ADST-II-CDRL-TF-9700197-A
August 15, 1997

ADST II

FINAL REPORT:

TERRAIN FIDELITY EXPERIMENT

DELIVERY ORDER

(CDRL AB02)

FOR: STRICOM
12350 Research Parkway
Orlando, FL 32826-3224

BY: Lockheed Martin Corporation
Lockheed Martin Information Systems
P.O. Box 780217
Orlando, FL 32825

Approved for public release; distribution is unlimited.

DTIC QUALITY INSPECTED 3

19980113 115

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	15 Aug 1997	FINAL

4. TITLE AND SUBTITLE
FINAL REPORT TERRAIN FIDELITY EXPERIMENT DELIVERY ORDER

5. FUNDING NUMBERS
N61339-96-D-0002

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Lockheed Martin Corporation
Information Systems Co.
12506 Lake Underhill Road
Orlando, FL 32825

8. PERFORMING ORGANIZATION REPORT NUMBER
ADST-II-CDRL-TF-9700197A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

NAWCTSD/STRICOM
12350 RESEARCH PARKWAY
ORLANDO, FL 32826-3224

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
A - Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***

This report describes an experiment to study intervisibility anomalies between a Dismounted Infantry (DI) entity and a Tank entity as they interact in a distributed simulation exercise. The experiment involves the application of non-real-time algorithms to compensate for differences in geometric terrain representations in the DI and Tank's synthetic environments. From the results of the experiment, we conclude that by modifying a remote entity's perceived visibility we can eliminate or reduce intervisibility anomalies and that this approach is an effective solution in many cases. Also, based on the results, we recommend a stand-alone, off-line tool which can be used to help correct, minimize, or avoid the intervisibility problem areas in dissimilar databases.

14. SUBJECT TERMS
STRICOM;ADTS-II;SIMULATION; TERRAIN;INTERVISIBILITY;DI;TANK

15. NUMBER OF PAGES
57

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

DTIC QUALITY INSPECTED 3

DOCUMENT CONTROL INFORMATION

Revision	Revision History	Date
-	Original release	4/15/97
A	Updates to incorporate IPT reviewer's comments	8/15/97
-		
-		
-		
-		
-		
-		
-		
-		
-		
-		
-		
-		

Table of Contents

1. Introduction 1
1.1. Background 1
1.2. Problem Space 1
1.3. Simulated Environment 2
1.4. Revisiting the Problem Space 3
2. Objectives 3
3. Approach 5
4. System Design 9
4.1. Hardware Configuration 9
4.2. Software Design 10
4.2.1. The Geometric Appearance Metric Algorithm 10
4.2.2. The Geometric Appearance Function Algorithm 13
4.2.3. Other Software Design Issues 14
5. Experiment Databases 15
6. Experiment Preparation 16
6.1. The Terrain Analysis Tool 16
6.2. The Environment Geometry Analysis Tool 18
7. Experiment Results Analysis 20
7.1. Scenario 1 20
7.2. Scenario 2 22
7.3. Scenario 3 24
7.4. Scenario 4 26
7.5. Scenario 5 28
7.6. Scenario 6 30
7.7. Scenario 7 33
7.8. Scenario 8 35
7.9. Scenario 9 37
7.10. Scenario 10 39
8. Other Interoperability Issues 42
9. Conclusions 42
10. Recommendations 44
11. Appendix 45

List of Figures

FIGURE 1: DIFFERENT VISIBILITIES AS A RESULT OF TERRAIN GEOMETRY 5
FIGURE 2: INCORRECT RESULT FROM ADJUSTING BOTH REMOTE ENTITY POSITIONS 6
FIGURE 3: CORRECT RESULT FROM ADJUSTING ONE REMOTE ENTITY POSITION 6
FIGURE 4: HOST PROCESS FLOW 8
FIGURE 5: TESTBED CONFIGURATION 9
FIGURE 6: MASKS FOR THE GAM 12
FIGURE 7: OCCULTING MASK COMBINED WITH THE MODEL MASK 12
FIGURE 8: OCCULTING MASK COMBINED WITH THE SHIFTED MODEL MASK 13
FIGURE 9: VIEWPOINT ATTITUDE INTEROPERABILITY ISSUE 14
FIGURE 10: REMOTE ENTITY VIEW CORRECTION 14
FIGURE 11: FORT BENNING MCKENNA MOUT DATABASE 15
FIGURE 12: TAT RESULTS FOR THE TWO MOUT DATABASES 17
FIGURE 13: TAT RESULTS OVERLAID ON MOUT DATABASE 17
FIGURE 14: EGAT RESULTS EXAMPLE 19
FIGURE 15: SCENARIO 1 EGAT RESULTS 21
FIGURE 16: SCENARIO 1 POSITIONS 21
FIGURE 17: TANK'S VIEW IN SCENARIO 1 22
FIGURE 18: SCENARIO 1 GAM RESULTS 22
FIGURE 19: SCENARIO 2 EGAT RESULTS 23
FIGURE 20: SCENARIO 2 POSITIONS 23
FIGURE 21: TANK'S VIEW IN SCENARIO 2 24
FIGURE 22: SCENARIO 2 GAM RESULTS 24
FIGURE 23: SCENARIO 3 EGAT RESULTS 25
FIGURE 24: SCENARIO 3 POSITIONS 25
FIGURE 25: TANK'S VIEW IN SCENARIO 3 26
FIGURE 26: SCENARIO 3 GAM RESULTS 26
FIGURE 27: SCENARIO 4 EGAT RESULTS 27
FIGURE 28: SCENARIO 4 POSITIONS 27
FIGURE 29: TANK'S VIEW IN SCENARIO 4 28
FIGURE 30: SCENARIO 4 GAM RESULTS 28
FIGURE 31: SCENARIO 5 EGAT RESULTS 29
FIGURE 32: SCENARIO 5 POSITIONS 29
FIGURE 33: TANK'S VIEW IN SCENARIO 5 30
FIGURE 34: SCENARIO 5 GAM RESULTS 30
FIGURE 35: SCENARIO 6 EGAT RESULTS 31
FIGURE 36: SCENARIO 6 POSITIONS 31
FIGURE 37: TANK'S VIEW IN SCENARIO 6 32
FIGURE 38: SCENARIO 6 GAM RESULTS 32
FIGURE 39: TANK'S VIEW OF DI DELTA POSITION IN SCENARIO 6 32
FIGURE 40: SCENARIO 7 EGAT RESULTS 33
FIGURE 41: SCENARIO 7 POSITIONS 33
FIGURE 42: TANK'S VIEW IN SCENARIO 7 34
FIGURE 43: SCENARIO 7 GAM RESULTS 34
FIGURE 44: TANK'S VIEW IN SCENARIO 7 OF DI DELTA POSITION 35
FIGURE 45: SCENARIO 8 EGAT RESULTS 35
FIGURE 46: SCENARIO 8 POSITIONS 36
FIGURE 47: TANK'S VIEW IN SCENARIO 8 36
FIGURE 48: SCENARIO 8 GAM RESULTS 37
FIGURE 49: SCENARIO 9 EGAT RESULTS 37
FIGURE 50: SCENARIO 9 POSITIONS 38
FIGURE 51: TANK'S VIEW IN SCENARIO 9 38
FIGURE 52: SCENARIO 9 GAM RESULTS 39
FIGURE 53: INDEPENDENT VIEW IN SCENARIO 9 OF DI DELTA POSITION 39
FIGURE 54: SCENARIO 10 EGAT RESULTS 40

FIGURE 55: SCENARIO 10 41
FIGURE 56: TANK'S VIEW IN SCENARIO 10 41
FIGURE 57: SCENARIO 10 GAM RESULTS 42
FIGURE 58: A SEQUENCE OF GAF DELTA POSITIONS IN A CONTINUOUS SIMULATION 43
FIGURE 59: SCENARIO 4 - DI HOST - DI MASKS 46
FIGURE 60: SCENARIO 4 - DI HOST - TANK MASKS 47
FIGURE 61: SCENARIO 4 - TANK HOST - DI MODEL MASK 48
FIGURE 62: SCENARIO 4 - TANK HOST - DI OCCULT MASK 49
FIGURE 63: SCENARIO 4 - TANK HOST - TANK MASKS 50

Acknowledgments

We would like to thank the members of the ADST II Terrain Fidelity Integrated Project Team: Traci Jones and Bill Wolfinger (STRICOM), Brian O'Connor and Tom Wilson (Lockheed Martin Information Systems - LMIS), Farid Mamaghani (IDA), and Paul Birkel (MITRE).

We would also like to thank the following engineering personnel from LMIS: Bob Ferguson, Bob Edwards, Steve McCarter, Bill Mayes, and Rick Zajac. Finally, we thank the Acusoft team for their database work.

List of Acronyms

DB	Database
DBG	Database Geometry
DBI	(Mission Function) Database Interface
DI	Dismounted Infantry
DIS	Distributed Interactive Simulation
EGAT	Environment Geometry Analysis Tool
FOV	Field of View
GAF	Geometric Appearance Function
GAM	Geometric Appearance Metric
LOD	Level of Detail
LOS	Line of Sight
MITL	Man in the Loop
MOUT	Military Operations in Urban Terrain
PDU	Protocol Data Unit
SAF	Semi-Automated Forces
SGI	Silicon Graphics, Inc.
TAT	Terrain Analysis Tool
VP	Viewpoint
VR	Vector Ranging

1. Introduction

This report describes an experiment to study intervisibility anomalies between a Dismounted Infantry (DI) entity and a Tank entity as they interact in a distributed simulation exercise. The experiment involves the application of non-real-time algorithms to compensate for differences in geometric terrain representations in the DI and Tank's synthetic environments. From the results of the experiment, we conclude that by modifying a remote entity's perceived visibility we can eliminate or reduce intervisibility anomalies and that this approach is an effective solution in many cases. Also, based on the results, we recommend a stand-alone, off-line tool which can be used to help correct, minimize, or avoid the intervisibility problem areas in dissimilar databases.

1.1. Background

Simulation attempts to present a characterization of something in the real world. When two individual simulator entities attempt to interoperate in the context of a distributed simulation, they must have a common set of characterizations. When these sets do not correlate, the outcome of the simulation may be adversely influenced. We refine this broad observation by focusing on what is of greater concern to us for this experiment, that is, interoperability between two man-in-the-loop (MITL) simulators. If two trainees wish to take part in the same distributed simulation, they need to experience the same simulated conditions. Those experiences may include any of the human senses. When two MITL simulator entities interact with each other, divergence from a common characterization of the real world in one simulator may lead to an outcome which may not have been probable in the real world. A simple example is a pair of nearly identical MITL simulators, with the exception that one includes aural cueing. One participant is able to hear his opponent and thus take different actions than the other participant's silent perception of the simulated world. Several other unequal perceptions of the simulated world in distributed simulation exercises are possible. For example, dissimilar representations of weather conditions, sensor effects, weapons modeling, and visual acuity are common in networked simulation exercises.

1.2. Problem Space

Rather than attempt to discuss all of the problems of interoperability between simulators participating in a distributed simulation exercise, we will narrow the problem space considerably by identifying a specific aspect of interoperability. The first culling step we will apply to the problem space is to eliminate non-MITL simulations or semi-automated forces (SAF's). Although this culling step eliminates many things which are important to simulation-based military training,

we are still left with a large problem space. SAF's are not included in our study since we are primarily concerned with MITL simulations with out-the-window (OTW) visual displays (e.g. helmet mounted displays, projection screens, CRT monitors).

We will further constrain our problem space by considering only pair-wise interoperability issues in distributed simulation. It is well known to the simulation community that several additional problems arise when more than two entities are participating distributed simulation exercise. These n^2 type issues are too broad to be adequately studied within the scope of this experiment. Certainly the lessons learned between two simulators can, in many cases, be applied to more than two simulators.

Given that we have only two, MITL simulator entities involved in the distributed simulation exercise, what issues are involved in their interoperation? Well, there are still many. For this effort we are only interested in the situations in which the simulator's entities visually perceive each other. Dissimilar visual presentations of the synthetic environment often has a negative impact on a simulation exercise outcome and, therefore, is of significant interest to the simulation community.

1.3. Simulated Environment

MITL simulators include a representation of the physical environment. In training simulators this representation of the physical environment is sometimes simply referred to as the terrain database; however, that term typically refers to all of the geometry and attributes of the environment being modeled. Other aspects of the environment, such as color, texture, and material type, are applied to the basic geometry. For consistency in our discussion, we will refer to the database geometry (polygons) as the DBG. References to the synthetic environment will include all aspects of the environment and not just the DBG.

If each MITL simulator has the same DBG, then interoperability issues should not result from the DBG itself. However, even in the situation when the DBGs are the same, the characteristics of the simulator can result in differing visual perceptions of the environment geometry. This is common in current simulators, and can be a result of differences in load management techniques, polygon blending, weather effects, or display characteristics between systems. In order to make our problem simpler to study, we will constrain our DBGs to single level-of-detail (LOD) geometry, with similar load management, blending, weather effects and visual displays.

When we consider the possibility of having two MITL simulators with different DBGs, we must have an idea of how and why they will be different. Suppose, we have two M-1 tank simulators. Why make the DBGs different if they are to

participate in the same geographic area during a distributed simulation exercise? Well, one major reason is that several different visual system hardware platforms are in use throughout the simulation community and standardization of a run-time visual data base format is not likely. Therefore, the use of different synthetic environments to represent the same geographic area will continue to be a source of interoperability anomalies in many distributed simulation exercises.

1.4. Revisiting the Problem Space

Now that we have constrained our problem space considerably, let us re-state the problem. We are concerned with interoperability between two entities participating in a distributed simulation, one simulating a DI and the other a tank. We are interested in the differences in the visual aspects of their simulations, specifically the visual anomalies induced as a result of the different terrain DBGs.

For the purpose of this experiment we will manipulate the tank's synthetic environment to contain less terrain geometry detail than the DI's synthetic environment. Since the DI's will operate in a higher fidelity environment we will consider the DI's DBG as the reference DBG. In other words, the DI's visual perception of the world will be considered true and only the tank's simulation will compensate for discrepancies between the two visual perceptions.

We also need to consider other factors that might cloud our observations. Non-geometric components of the synthetic environment representation can certainly differ. Examples of these components include color and texture. By applying the same texture and color attributes and tuning the display systems, we will avoid the issues associated with the non-geometric aspects of the environment.

The issues related to the constrained problem are now becoming more manageable. Using the DI's "true" representation of the DBG we will determine what percentage (0-100%) of the DI's geometry is visible from the tank's perspective in the DI's synthetic environment. After transmitting the DI's "percent visible" data to the tank's simulator, the tank's host computer will attempt to adjust the DI's position in the tank's environment so that the DI has the same relative visibility in both synthetic environments.

2. Objectives

In accordance with our previously stated assumptions, we define our objective of this experiment. Our objective is to study the intervisibility problems between a DI and a tank when interoperating on different DBGs. Methods shall be defined to quantify an entity's visibility or appearance (the Geometric Appearance Metric, or

GAM), and to modify the result of that quantification (the Geometric Appearance Function, or GAF). These methods will be applied during the execution of the simulation. In our implementation, the application of these methods do not occur in real-time and involve stationary entities. The study concentrates on geometric visibility and not detectability or identification.

Beyond the primary task of meeting the objective, there are several other tasks. We categorize these tasks into those which are required or proposed in the approach (secondary tasks) and those are deemed necessary because of the implementation (tertiary tasks).

Secondary tasks:

- Use of the Fort Benning McKenna MOUT DB - A customer requirement to use a specific DB.
- Generation of the tank version of the MOUT DB - A customer requirement to derive a less dense DB from the required DB.
- Use of two different image generators - A customer requirement to ensure that the algorithms proposed for the experiment are not dependent upon a specific IG.
- Creation of the GAM and GAF algorithms - Proposed algorithms as part of the interoperability solution.
- Interim and final demonstrations of the experiment - A customer requirement to demonstrate the algorithms designed for the experiment.

Tertiary tasks:

- Creation of a DI entity for the host simulator - Proposed host modifications to support a DI entity.
- Enhancement of the mission functions package - Proposed host modifications to support orientation of models and their articulated parts.
- Creation of several PDUs for communication between hosts - Proposed host modifications to exchange the data necessary for the experiment.
- Creation of the EGAT algorithm - A proposed, off-line algorithm to help find scenario positions.

3. Approach

When the intervisibility between two entities is different because of different DBGs, two straightforward solutions exist. Move either or both entities so that the intervisibilities are equal, or modify the DBGs so that the difference does not occur. Applying the latter throughout the database may result in the DBGs no longer being dissimilar. While this is a possible solution, the experiment concentrates on the former approach.

In

Figure 1, the DI and tank cannot see each other in the DI's database, and have partial visibility of each other in the tank's database. Figure 2 shows the result of moving the remote entity in each simulator. In the DI database, the tank is moved to make it visible so that the visibility computed in this new position will be (roughly) equal to the visibility in the tank database. In the tank database, the DI is moved so that it is not visible. It should be apparent that the situation has now been reversed, but the problem is the same. One entity still has an unfair advantage due to the difference in DBG.

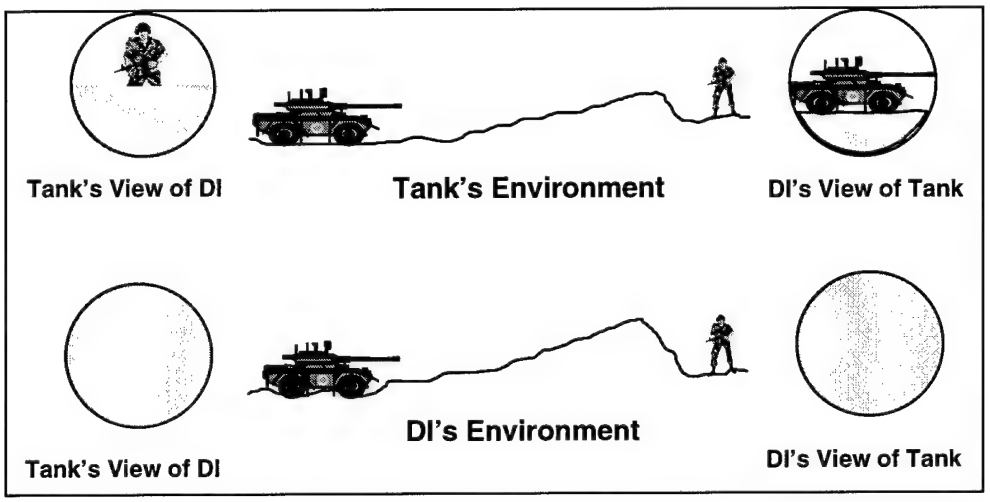


Figure 1: Different Visibilities as a Result of Terrain Geometry

For the purposes of our experiment we take the view that the DI's database is higher fidelity and is closer to the real world than the tank's database. Therefore, we will consider the visibilities computed in the DI's simulator to be correct and any discrepancies between the visibilities in the two simulators will be compensated for in the tank's simulator only. When we apply this rationale to the previous example, the DI will retain its original perception of the tank, and the tank will have a modified perception of the DI as shown in Figure 3. Now the percieved visibilities are equal and the interaction is more fair.

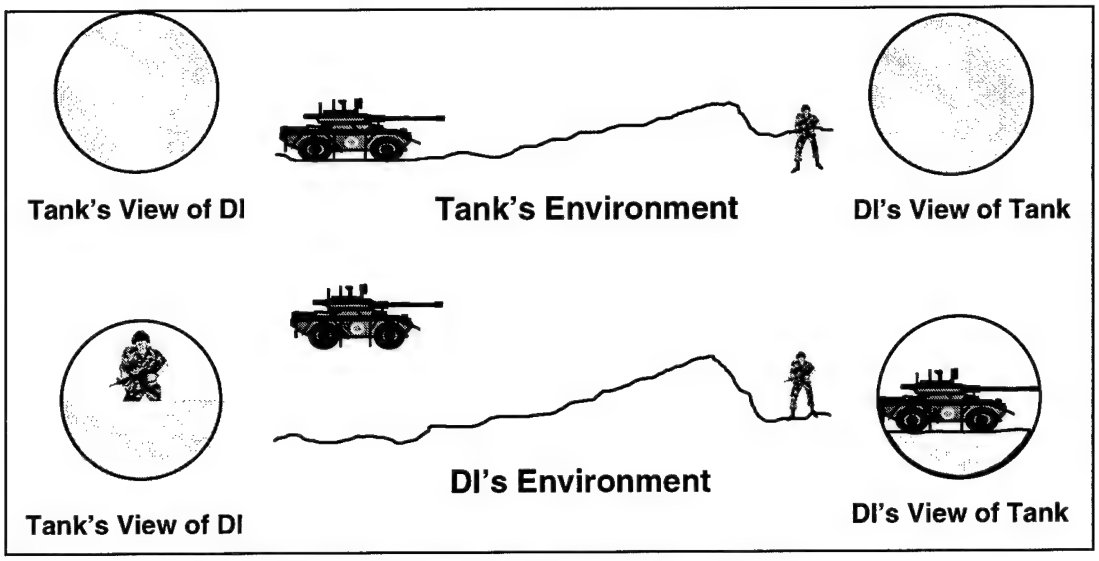


Figure 2: Incorrect Result from Adjusting Both Remote Entity Positions

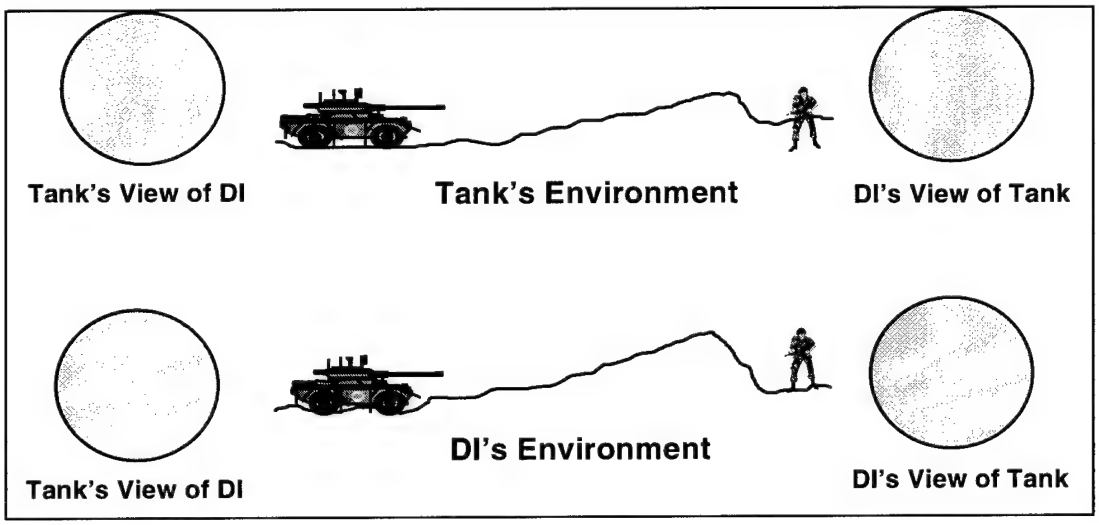


Figure 3: Correct Result from Adjusting One Remote Entity Position

We can generalize the relationship between the two databases in the event that one is not of a higher fidelity. As long as we choose one simulator's database to be the master, only the other simulator needs to compensate for visibility anomalies. Regardless of the choice, if the simulator without the master database is able to move the remote entity so that the percieved visibility of the entity's are mutual, the interaction between the two entities will be more fair.

Geometric visibility of an entity is quantified by point sampling the entity's geometry with a vector ranging (or line-of-sight) function. The frequency of this sampling is proportional to the simulator's image resolution. From the sampling results, the visibility can be quantified. This is expressed as a ratio of (a) the number of unobstructed samples hitting the entity to (b) the number of samples hitting the entity. This expression is referred to as the Geometric Appearance Metric (GAM).

Each simulator computes two GAMs. The first is the GAM for the local entity's view of the remote entity. The second is the GAM for the remote entity's view of the local entity. One simulator compares its first GAM to the other simulator's second GAM. A difference in those GAMs requires the simulator, on which the entity is remote and on which the database is not the master, to reposition the remote entity such that the new position would result in a GAM equal to the remote host's GAM. This action is referred to as the Geometric Appearance Function (GAF). Since computation of the GAM is not a real-time operation in our implementation, it is expensive to re-compute the GAM for each new position under consideration. The results of the GAM computations can be reused if the movement is restricted to a plane parallel to the image plane. We take this view with the intent that the GAM computations could be redesigned for a real-time environment.

Figure 4 shows a simulator's process flow. In our experiment, the DI simulator initiates the application of the algorithms. In response to a control button, the DI simulator requests VP data from the tank host. It then sends its own VP data to the tank simulator. It can begin the GAM computations for the tank entity. When that operation is finished, the DI simulator waits for the tank VP data if it has not already arrived. The DI simulator then computes the GAM for the DI entity from the tank's VP. This GAM result is sent to the tank simulator. When the tank simulator's GAM result is received, the DI simulator can begin the GAF computations if the GAMs differ. In our experiment, the DI simulator does not execute the GAF since it has the master DB. The GAF is executed in the tank simulator if the GAMs differ. The tank simulator may reposition the DI entity if necessary.

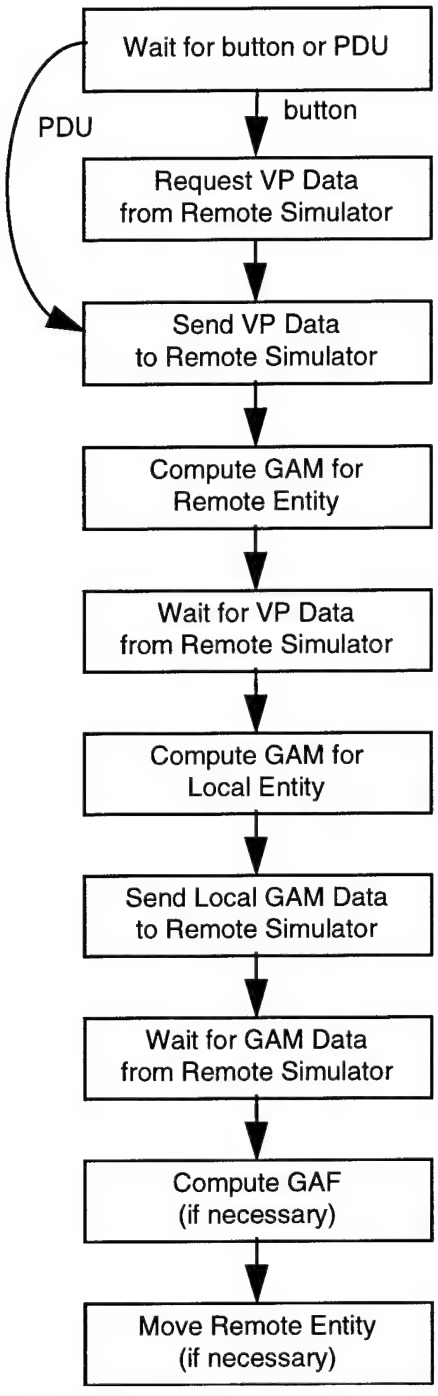


Figure 4: Host Process Flow

Since the DI host does not execute the GAF, it does not need to compute the GAM for the tank entity. With that in mind, the tank host does not need to compute the GAM for the tank entity from the DI's VP since the DI host will not use the result. This optimization is omitted from our implementation. It may appear that the VP data exchange is not necessary since it could be derived for

the remote entity. We will show in our algorithm discussion that the exchange is necessary.

4. System Design

The experiment leveraged available resources as much as possible. Loaned equipment and existing software were used in order to keep the cost down.

4.1. Hardware Configuration

The testbed for the experiment consisted of the following: a Sun Sparc workstation for the DI simulator, a Sun Ultra workstation for the tank simulator, a Real 3D Pro 1400 image generator for the DI simulator, an SGI Max Impact image generator for the tank simulator, and a flybox input device for each simulator. Figure 5 illustrates most of the equipment in the testbed. The workstations communicated with one another by means of DIS 2.04 PDUs over an ethernet network connection. The DI simulator communicated with the Real 3D Pro 1400 through a wide SCSI. The tank simulator communicated with the SGI Max Impact using the previously mentioned ethernet network using sockets. Each host interfaced to its flybox through a serial port (not shown in the figure).

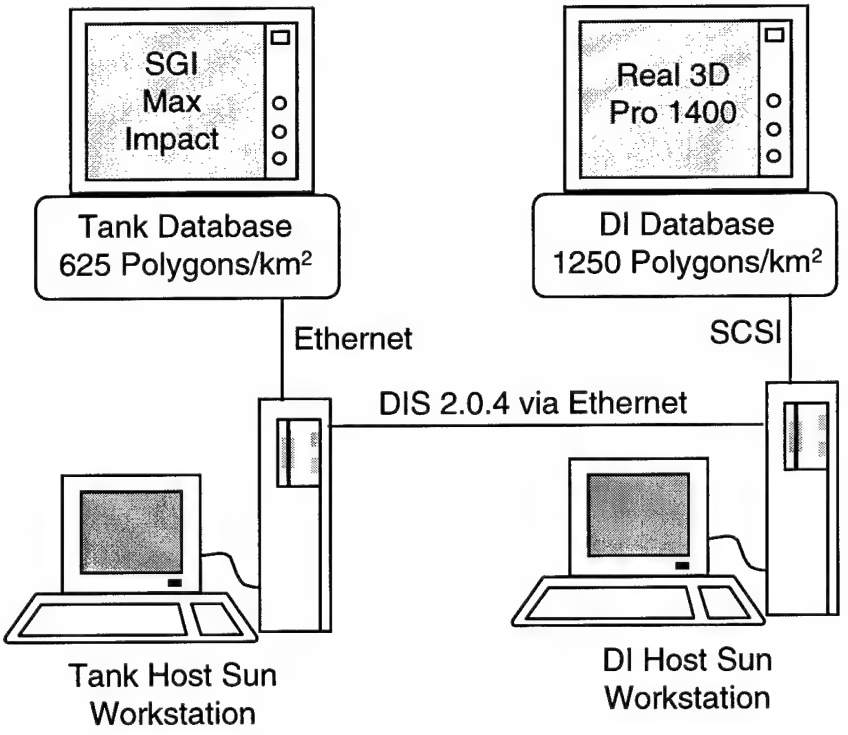


Figure 5: Testbed Configuration

4.2. Software Design

The Reconfigurable Host Simulator (referred to as "the host" in the remainder of this report) is existing host software which is capable simulating various entities and interfacing to different IGs. The host already supported a tank entity. As identified in the experiment objectives, minor additions were necessary to support a DI entity.

One of the notable features of the host is a mission functions package which allows consistent results when interrogating the databases since both hosts will be executing the same operations. The mission functions package has a Database Interface (DBI) layer, which performs the vector-face intersection primitive, and this layer is incorporated into some other tools (discussed in Section 6). Additions were necessary to properly orient the entity models and their articulated parts.

As stated, the host can interface to several IGs. An interface already existed for the Real 3D Pro. Minor additions were necessary for the SGI Max Impact. The host can interact with other hosts through DIS 2.04 PDUs. Minor additions were made to the host to support some new PDUs for the experiment.

The GAM and GAF algorithms were also incorporated into the host. Although the algorithms were as isolated from the rest of the host as possible, there was some impact to the design because of the host characteristics. Because of the dependence on the mission function package and requirements within the host to maintain a real-time environment, a monitor algorithm was developed to allow the GAM and GAF algorithms to execute one iteration per field. This limitation is not necessary within the GAM and GAF algorithms themselves.

For the GAM and GAF algorithms, a new mission function, which was related to vector ranging, was created. A special vector ranging function was designed to return two results: (1) the range to a specified model if the vector hit the model, and (2) the range to the closest database feature besides the specified model if anything was hit up to a specified distance. The existing vector range could return either of the two mentioned ranges, but not both.

4.2.1. The Geometric Appearance Metric Algorithm

The GAM quantifies the geometric appearance, or visibility, of an object on a simulator so that this quantification can be compared to that on another simulator. Our implementation of the GAM algorithm point samples the environment in order to determine what can be seen. Other implementations are possible, and they should not alter the analysis of the results of the GAM concept.

The GAM relies on vector ranging (VR), or line-of-sight (LOS), as an underlying tool. The LOS and VR terms are commonly used interchangeably in the simulation community, but it is really VR in which we are interested. The significant difference is that we are interested in what lies along a vector up to a certain range. The vector may not be along the boresight (which LOS usually is), and we do not care what is beyond the range (and LOS does).

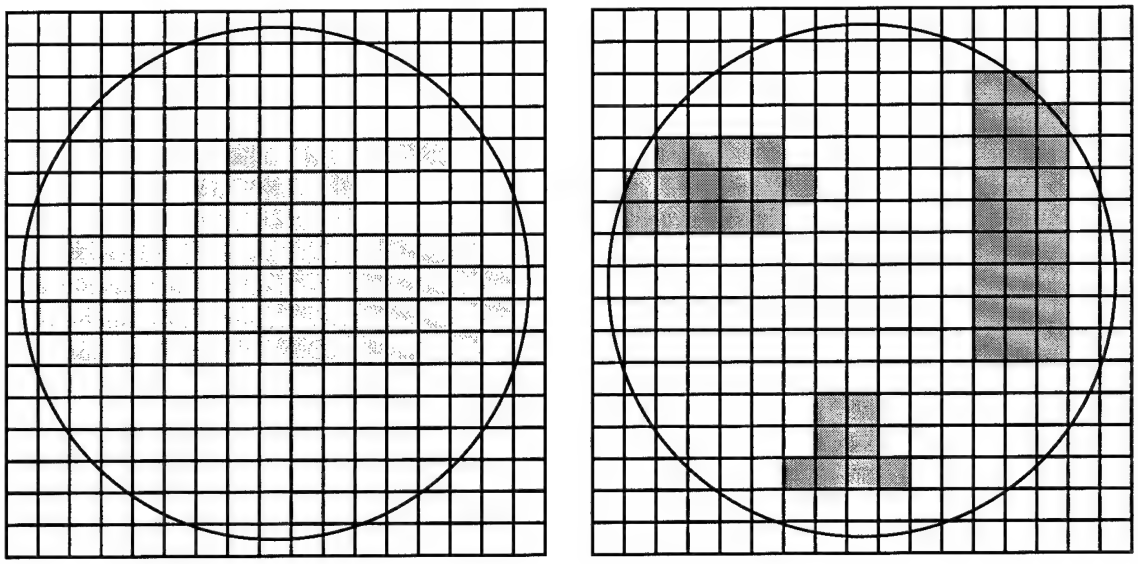
The results of these VR samples tell us (1) if the target is hit by the vector (and is thus potentially visible along it), and (2) if something occults the (potentially visible) target. From these results we can conclude what percentage of the target-striking vectors are not occulted by something. Vectors which do not strike the target at all are not useful in computing the visibility percentage, but are key to determining if the target can be moved in order to change the visibility percentage. This process is the Geometric Appearance Function (GAF) and is discussed in Section 4.2.2

The GAM requires the following input parameters: (1) the field of view and sampling resolutions of the display device, (2) the viewpoint position and attitude, and (3) the target position and size. Our implementation of the GAM attempts to relate the visibility quantification to the displayed image. All of the parameters are required to create the sampling vectors.

Our implementation of the GAM algorithm consists of several simple geometric calculations. From the field of view and sampling resolutions, the distance between samples is determined. This distance will let determine how many samples to take around the entity position. A sample consists of a vector origin, which is always the viewpoint position, and a vector direction. The directions of the samples will vary, but will be oriented generally toward the target entity. Each vector will strike the sphere which bounds the entity. The GAM will relate how many of these vectors which strike the entity itself (and not just the bounding sphere) are not occulted by some intervening geometry.

The result of each sample will relate two things: (1) if the sample intersects the target entity's geometry, and (2) if the sample intersects any other geometry. Each sample vector has a length limited to the distance to the entity plus the radius of the sphere bounding the entity. The results will also denote the distances to the intersection points, when they exist. As an example, suppose a soldier stands one foot away from a building. All samples would show intersections with other geometry, but they would be further away than the intersections with the soldier's geometry.

The results of the samples are recorded in two masks. The intersections with the target entity's geometry are stored in the Model Mask. The intersections with the other geometry are stored in the Occulting Mask. Figure 6 shows a pictorial representation of an example.



Model Mask Occulting Mask

Figure 6: Masks for the GAM

The GAM is computed by comparing the results in the two masks. The GAM is simply the percentage of samples striking the entity which do not have an occulting geometry (i.e. an intersection distance in the occulting mask which is closer than the intersection distance in the model mask). Figure 7 shows a pictorial representation of combining the two masks.

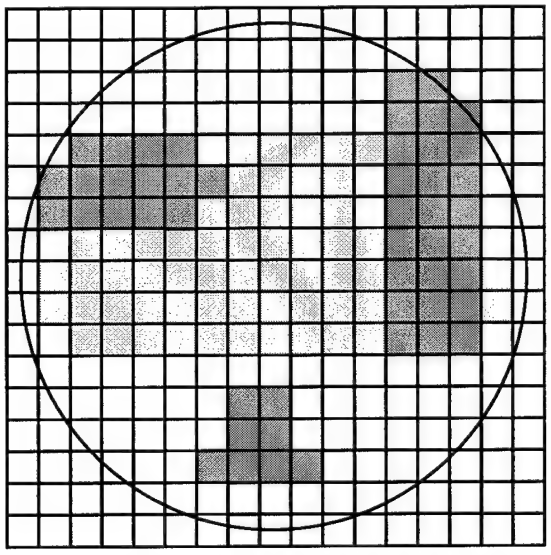


Figure 7: Occulting Mask Combined with the Model Mask

Simulator A computes the GAM for a target and compares it to the GAM computed by the target's simulator (B). In order for simulator B to compute the

GAM for itself from simulator A's viewpoint, it requires similar parameters for the GAM from simulator A.

4.2.2. The Geometric Appearance Function Algorithm

The objective of the GAF is to change the target's appearance so that a new application of the GAM would result in a quantification close to that specified by the target's host simulator. Our implementation of the GAF consists of changing the target entity's position so that its relationship to the other geometry in the environment will be different. The changing of the target's position is local to the simulator, and no other simulator will be aware of such an adjustment.

The GAF algorithm has two stages. The first is an extension of the GAM. This involves more sampling around the target position. The second uses the extended sampling results as an aid in moving the target in order to obtain the desired quantification.

Figure 7 shows the result of overlaying the occulting mask on the model mask. Of the original 73 samples, only 58 are visible. Thus, the GAM is 58 / 73 = 79%. Figure 8 shows that by moving the model mask downward one pixel, the GAM is increased to 62 / 73 = 85%. Moving the model mask may result in the need of information outside the model radius.

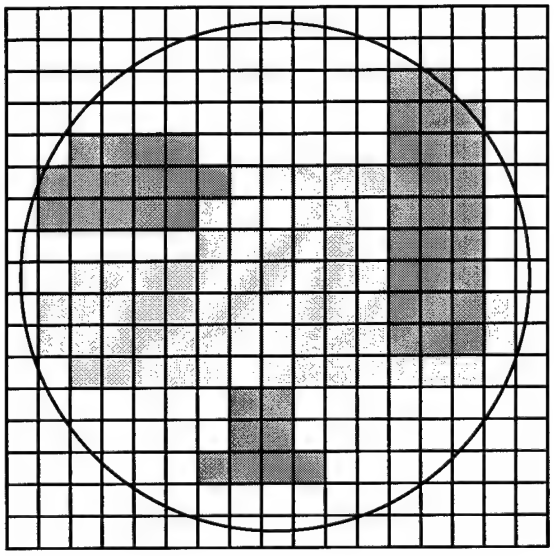


Figure 8: Occulting Mask Combined with the Shifted Model Mask

As previously stated, the first stage of the GAF is to perform more sampling. We do this by increasing the radius of the circle enclosing the model. The amount by which to increase the radius, which we refer to as the threshold radius, is difficult to determine. We may not want to move the model too far. Also the larger the radius, the longer it will take to compute the expanded occulting mask.

4.2.3. Other Software Design Issues

For the experiment, the GAM and GAF algorithms run on demand. A flybox control starts the execution of the algorithms on one host. This host sends a data query PDU to the other host requesting viewpoint data. It also sends its viewpoint data. An interesting interoperability issue arises at this point.

The viewpoint position and attitude sent by the remote host may not correspond to the viewpoint position and attitude of the remote entity on the local host due to database geometry differences. Figure 9 illustrates how an entity's view direction can be vastly different in the two simulators.

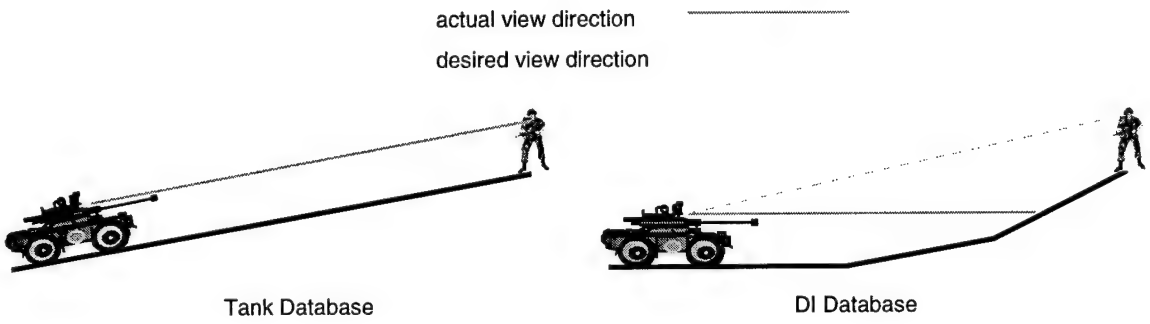


Figure 9: Viewpoint Attitude Interoperability Issue

Figure 10 shows a possible image for the scenarios in Figure 9. In the tank's simulator, the center of the screen corresponds to a point to the left of the DI. In the DI's simulator, the DI is off screen. In order to compare GAM results, the tank's view direction needs to be adjusted to a point that brings the DI into view.

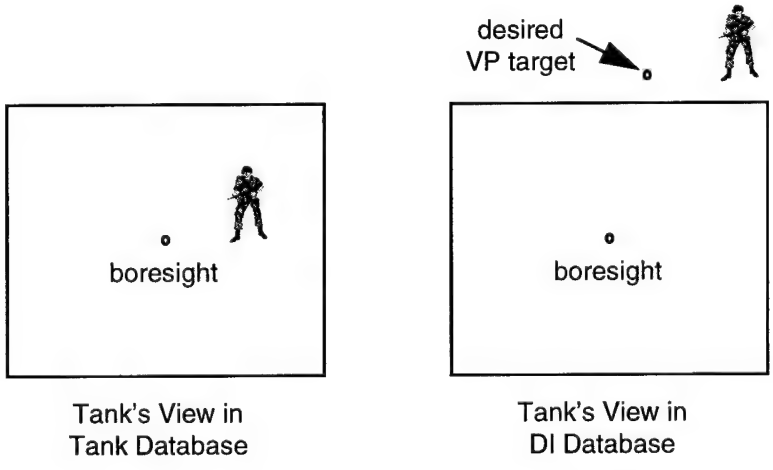


Figure 10: Remote Entity View Correction

So, the remote host (in this example, the DI host) computes the viewpoint position for the position and attitude for the remote entity. The viewpoint direction does not necessarily correspond to the resulting line of sight. Instead, a desired VP target point will be computed using information obtained from the other host. Instead of each host sending a VP position and attitude, it will send an offset from the target entity which is along the line of sight. This offset will be used by the remote host to determine the line of sight. This adjustment may not be exact, but it is much better than making no adjustment.

5. Experiment Databases

To assist in demonstrating the algorithms in the experiment, an existing database was chosen. This database is a 2 kilometer by 2 kilometer region surrounding the Fort Benning McKenna MOUT. Figure 11 shows a downward view of the database from above the MOUT area. In the figure, the lower left corner has (X, Y) coordinates of (500, 500) and the upper right corner has coordinates (2500, 2500).

Figure 11: Fort Benning McKenna MOUT Database

From the original database, a database was made with the interiors of all but three buildings removed. This database was used in the DI host. From this database, a less dense database was derived for the tank host. The density reduction was in a corridor containing the middle third of the database. The

terrain density was made approximately half in the affected region. In this database, the interiors of remaining buildings were removed. Additional information concerning the databases can be obtained in the database description document[1].

The databases and all models were represented in the .FLT format. The tank model was an M1A2 tank with an articulated turret and gun barrel. The DI model consisted of three independent representations, which were a standing DI, a kneeling DI, and a prone DI. Host modifications were made to handle the three models (as opposed to a single, articulated model).

6. Experiment Preparation

To perform the experiment, test cases were required. To alleviate the problem of finding them manually, some tools were developed. Once candidate positions were found using the tools, the experiment algorithms were run so as to ensure that there was a variety of test cases.

6.1. The Terrain Analysis Tool

The Terrain Analysis Tool (TAT) was designed to compare terrain elevations from two databases using a grid of sample points. Each database was sampled and the differences in elevation were displayed graphically.

The TAT uses the same DBI which is used by the host's mission functions. Since the DBI could only load one database at a time, the TAT consists of two parts: one which computes the terrain elevations at the sample points in one database and writes them to a file, and another part which compares the results in the two files.

The databases were sampled at 10 meter intervals. In Figure 12, green and yellow areas show where the DI database was higher at the sample point than the tank database. Blue and purple areas show where the tank database was higher than the DI database. A parameter to the display program could be set to identify where a difference was more than a desired threshold. In this example, a 3 meter difference was chosen. The yellow and purple areas denote where the difference in elevation was more than 3 meters. The green and blue areas have the color intensities scaled according to the difference in height up to the threshold (i.e., differences near 0 meters have the lowest intensity, and differences near 3 meters have the highest intensity).

[1] "Visual Database Description Document for the Terrain Fidelity Delivery Order McKenna MOUT Database," ADST-II-MISC-TF-9700201, December 16, 1996.

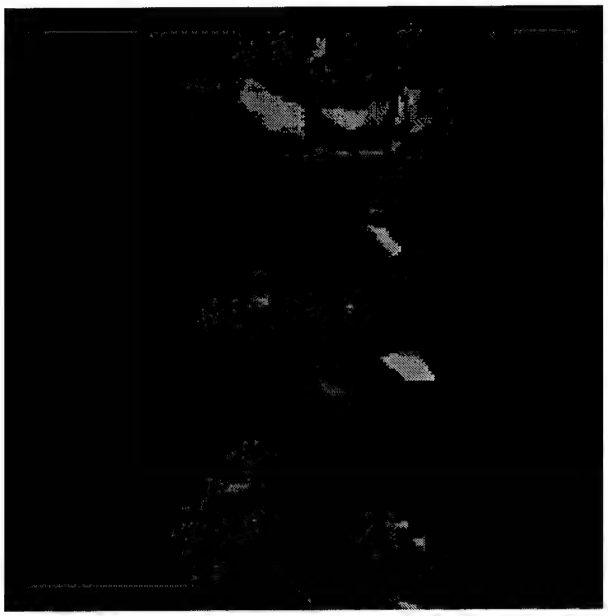


Figure 12: TAT Results for the Two MOUT Databases

The non-zero differences can be mapped onto the MOUT database image to show where the affected areas are. This is shown in Figure 13. There are no terrain differences within the area immediately around the buildings. Note that the buildings have interior differences, but the figure only shows terrain differences. The left and right sides of the image have no differences because the tank database was not modified in these areas with respect to the DI database.

Figure 13: TAT Results Overlaid on MOUT Database

Scenario positions involve either at least one entity being in a colored area or both entities being on opposite sides of a colored area. Just because the elevation varies in a particular area does not mean that the lines of sight will have a difference in occlusion. Further analysis is required to determine suitable scenario positions.

6.2. The Environment Geometry Analysis Tool

The Environment Geometry Analysis Tool (EGAT) was designed to assist in locating positions with intervisibility interoperability problems as a result of different DBGs. The tool examines points clustered around two positions for differences in lines-of-sight. Such differences in lines-of-sight may be the source of interoperability problems. These positions must be further investigated with the algorithms which were added to the host.

Like the TAT, the EGAT consists of two parts. The Data Generator module loads one database and processes test cases for that database. The results are written to a file for further processing by the Data Analyzer module. A test case consists of several data. The first parts of the data are two X and Y database positions (one for the DI and one for the tank). The Z for each position is computed from a DBI function. A delta Z for each position indicates at what height above the terrain Z the viewpoint is actually located. Two more parameters are the distance between generated sample points and the range limit from the initial position (i.e. how far to extend the sample points).

The Data Generator module iterates the different possible X and Y positions for the DI and the tank. It computes the height above terrain for the positions and increases the Z by the appropriate delta for the entity. Then an intervisibility test is performed between the two points. The results of the comparisons are written to an output file along with the X, Y, and Z for each position. The Data Generator module is run for the other database.

Then the Data Analyzer module reads the two generated files and creates a list of inconsistencies between the intervisibility results. An inconsistency is a result of either (1) the occluding geometry in one database is absent or different (in position or shape) in the other database, or (2) the occluding geometry has an identical counterpart in the other database, but one or both entity positions are different, and thus the vector between the two entities in one database is different from the vector between the two entities in the other database.

Figure 14 gives a 2D example illustrating the Data Generator algorithm results. The DI is at position *b* and the tank at position *e*. The other positions are generated from the parameters. For simplicity, only positions on each side of each entity are generated. The two databases have different geometries in the

areas of interest. In the figure, the difference is shown by a face with a different size and orientation. For each pair of entity positions, a solid line shows that there is no occlusion, and a dashed line shows that there is occlusion. The results matrix denotes that two points are occluded with a 0 entry, or not occluded with a 1 entry.

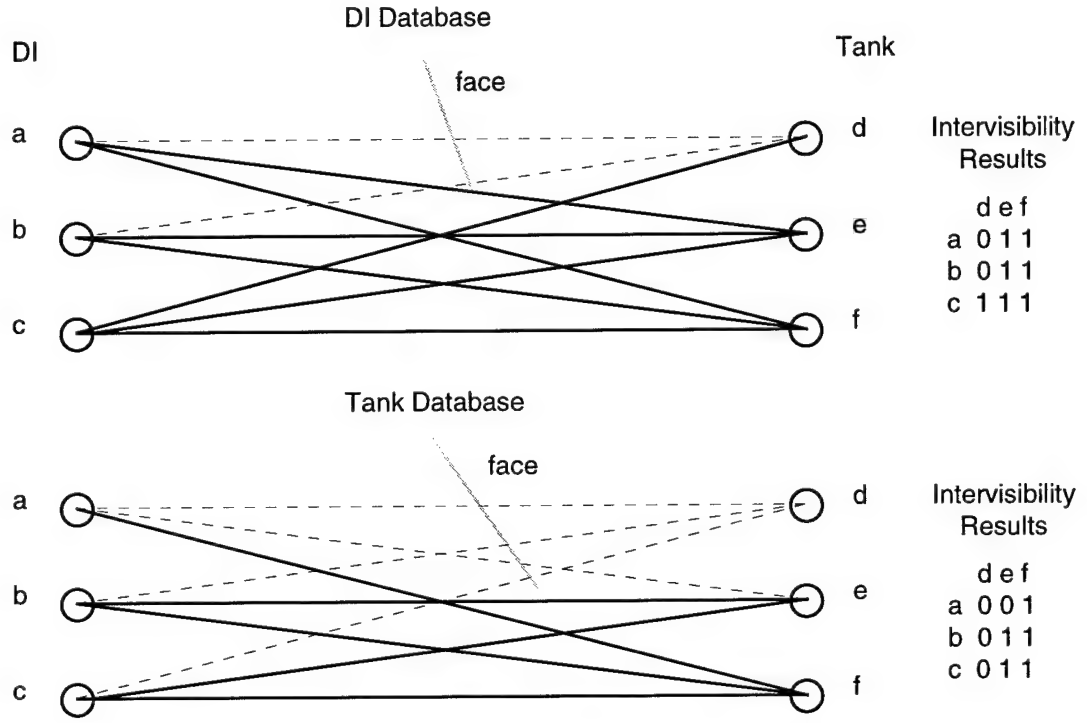


Figure 14: EGAT Results Example

The matrices for the two databases differ in entries (a, e) and (c, d). The results would indicate that interoperability problems are likely to occur when the tank is between *d* and *e*. The tool could be run again with a different start point and/or a smaller distance between samples in order to improve on the suggested locations.

Note that the EGAT sampling is a point-to-point sampling. Models are three-dimensional volumes that occupy a two-dimensional area in the image plane. While there may not be a line-of-sight difference between the two model viewpoint positions, there may be a line-of-sight difference between one viewpoint and another point on the target model. This will be exemplified in Scenario 9. This does not invalidate results which show differences. It may invalidate those that do not. Also, because the sampling is point-to-point, differences may occur between samples and may not be shown by the samples themselves.

7. Experiment Results Analysis

The DI database is considered to be the master database for the GAF computations. The threshold radius supplied to the GAF by the DI host is 0 so that no GAF computations are performed even though the GAMs might vary. The threshold supplied to the GAF by the tank host is twice the radius of the appropriate DI model (standing, kneeling, or prone), which is equivalent to the height (or length) of the model. This value was randomly chosen.

Each scenario has the EGAT results for the positions for the entities. For simplicity's sake, the input to the EGAT Data Generator had the distance parameter equal to the range parameter for each entity in order to generate a small set of results. So each test case has a 9x9 grid of results. The DI's test positions are labeled A-I, with position E being the input point to the EGAT algorithm. The tank's test positions are labeled J-R, with position N being the input point to the EGAT algorithm. If there is a vector ranging difference between the two databases, then a 1 is shown in the grid. Otherwise, a dash appears. Note that these results are not indicating occulting or non-occulting. A 1 shows that one database has occulting, the other does not.

To the right of each grid are points representing the data positions. The x-coordinate increases left-to-right, and the y-coordinate increases bottom-to-top. The (X, Y) database coordinates are given for the input points to the EGAT and the positions for the entities when the GAM algorithm was run. Points with vector ranging differences, if any, are enclosed in a shaded region. The actual positions for the entities are shown with colored boxes. The DI's position is represented by a blue box, and the tank's with a red box.

A top-down view of the database is shown with each entity's position marked by its respective colored box. Images are shown for the tank's view of the DI in the tank simulator. The GAM results are shown for each entity for each simulator. Examples of the model masks and occulting masks are given in the Appendix for scenario 4. The masks were generated for the DI views of the tank even through they were not necessary. Some scenarios show images of the DI in its GAF delta position.

7.1. Scenario 1

The first scenario was a control case in which each entity has complete visibility of the other. The DI and tank were on opposite sides of the lake near the edge of the database area. Figure 15 shows the EGAT results for the input positions. The DI's input position was E = (2405, 975) and the tank's was N = (975, 1035). One test position was generated in each direction at a distance of 5 meters. So, for the DI, position A = (2400, 970), B = (2405, 970), C = (2410, 970), D = (2400, 975), etc. The results matrix shows that there were no line-of-sight differences

between any pair of test points.

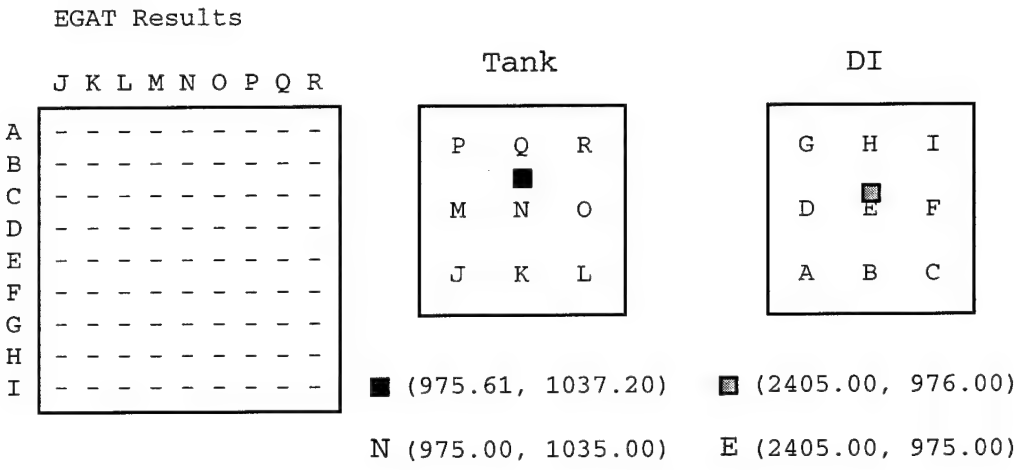


Figure 15: Scenario 1 EGAT Results

Figure 16 shows the entity positions from a top-down view of the database. Referring to Figure 13, it is apparent that there are terrain differences between the two entities. The terrain differences do not cause any intervisibility problems between the entities' positions. If the entities were to approach each other, the differences in terrain geometry might introduce intervisibility problems.

Figure 16: Scenario 1 Positions

Since neither database had any occulting faces between the two positions, each entity had complete visibility of the other. Figure 17 shows the image as seen by the tank in the tank's simulator.

Figure 17: Tank's View in Scenario 1

Figure 18 shows the entity positions in each database and the GAM results. In this case, each entity had the same elevation in each database. The GAMs were equal and no action was necessary by the GAF.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	2405.00	976.00	107.33	34 / 34	975.61	1037.20	104.23	17 / 17
Tank Host	2405.00	976.00	107.33	34 / 34	975.61	1037.20	104.23	17 / 17

Figure 18: Scenario 1 GAM Results

7.2. Scenario 2

The second scenario was also a control case in which each entity could not see the other. The DI was near the MOUT area and the tank was about 0.5 km away. Figure 19 shows that there were not any line of sight differences between the sample points.

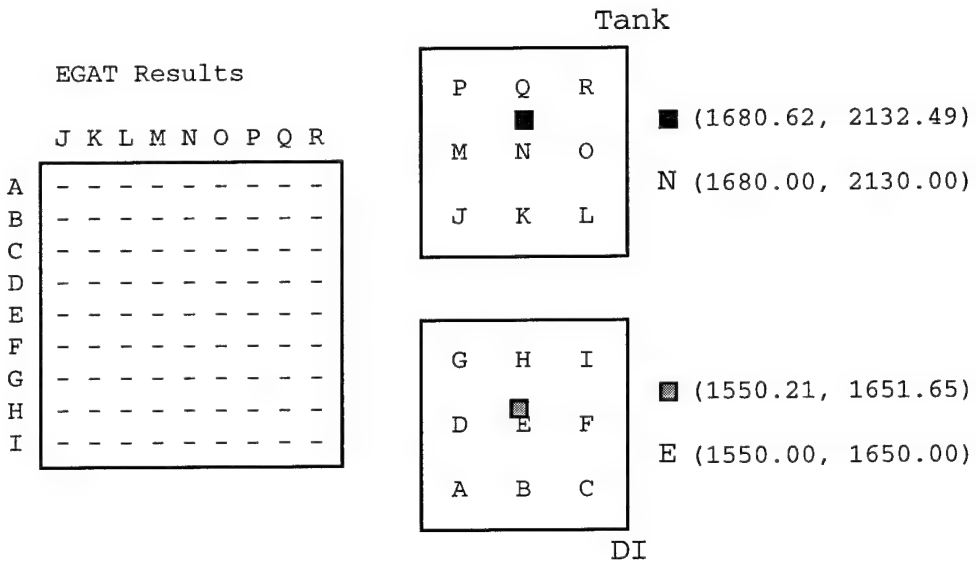


Figure 19: Scenario 2 EGAT Results

Figure 20 shows the entity positions in the top-down view of the database. When referring to Figure 13, there are terrain differences between the two positions.

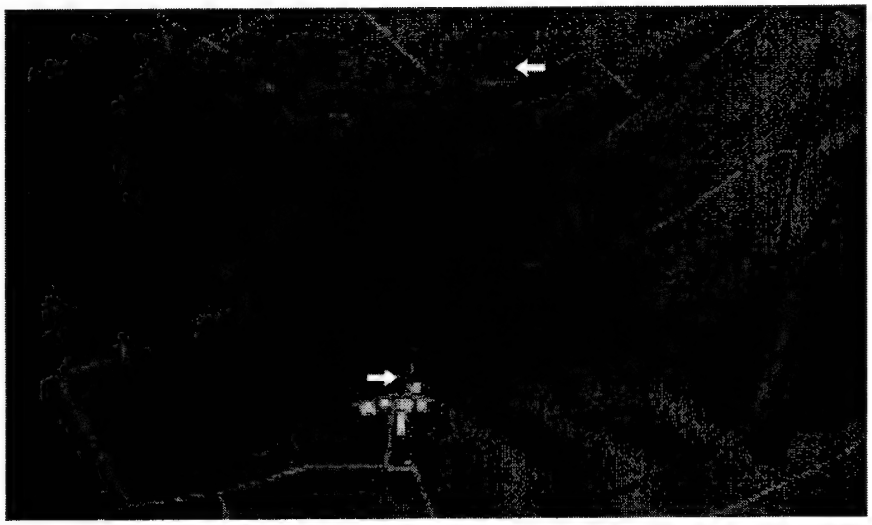


Figure 20: Scenario 2 Positions

Each database had occulting faces between the two entities, and so each entity had no visibility of the other. Figure 21 shows the tank's view in the tank's simulator. Of course, the DI is not visible in the image.

Figure 21: Tank's View in Scenario 2

Figure 22 shows that the GAMs were equal and so no action was necessary by the GAF. In this scenario, the distance to the occulting feature was significantly different in the two databases. In the DI database, the occulting feature was about 70 meters from the DI. In the tank database, it was about 150 meters from the DI. This was mostly attributed to the tank's smaller Z in the tank database since the terrain was identical near the DI. It would seem that moving the tank along the line of sight might alter the GAM equality. This concept is further discussed in some subsequent scenarios.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1550.21	1651.65	131.04	0 / 239	1680.62	2132.49	118.46	0 / 120
Tank Host	1550.21	1651.65	131.04	0 / 253	1680.62	2132.49	115.87	0 / 97

Figure 22: Scenario 2 GAM Results

7.3. Scenario 3

In scenario 3, the DI was near the MOUT area and the tank was 0.5 km away. The EGAT results in Figure 23 show that there is a good chance of finding a test case if each entity is within the shaded regions containing the vector ranging differences.

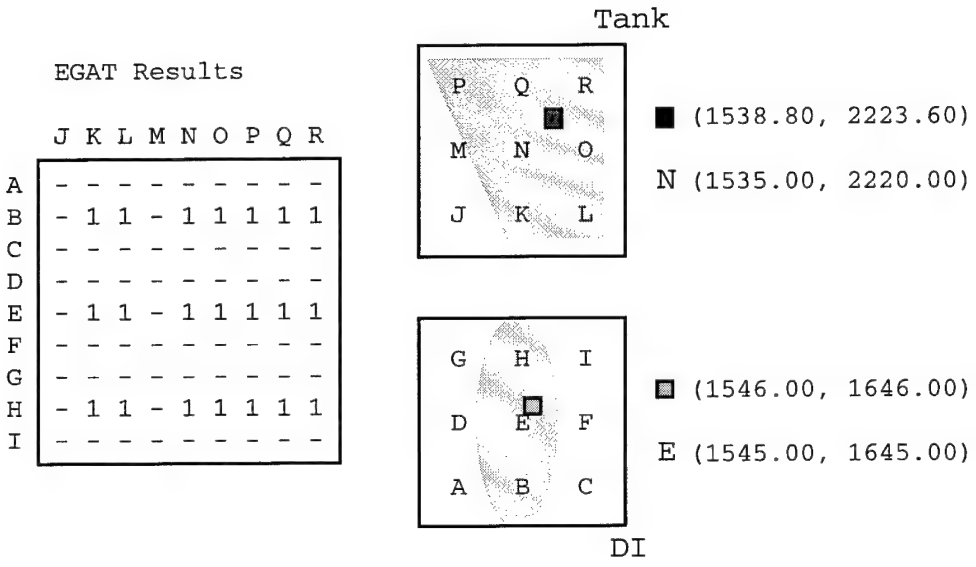


Figure 23: Scenario 3 EGAT Results

Figure 24 shows the entity positions in the top-down view of the MOUT database.

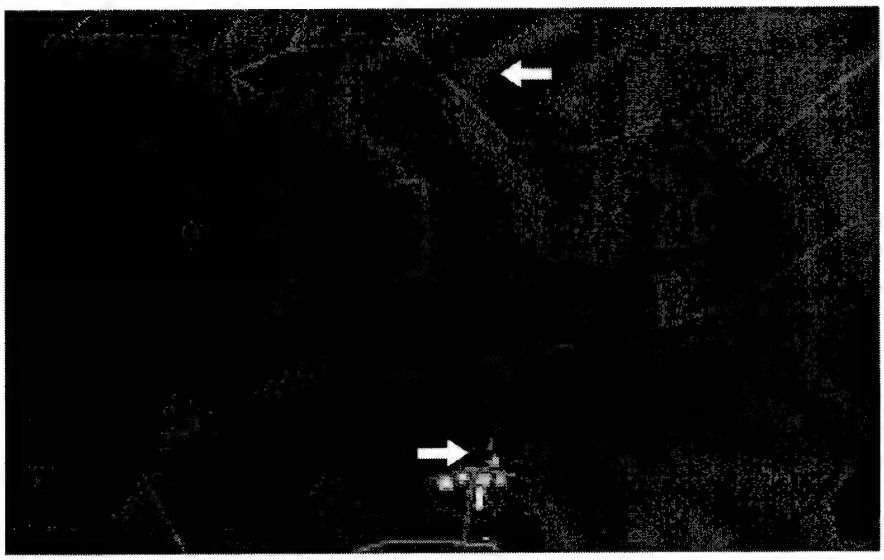


Figure 24: Scenario 3 Positions

In the DI database, each entity could not see the other. In the tank database, the entities had approximately 50% visibility of each other. The tank's view of the DI in the tank simulator is shown in Figure 25.

Figure 25: Tank's View in Scenario 3

Figure 26 shows that the GAMs were not equal, and so the tank host applied the GAF. The resulting delta position was 5 pixels left and 15 pixels down, which had a corresponding (X, Y, Z) of (0.137124, 0.003107, -0.739829). When the DI was translated by the delta position, it was no longer visible to the tank.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1546.00	1646.00	131.10	0 / 178	1538.80	2223.60	123.15	0 / 91
Tank Host	1546.00	1646.00	131.10	101 / 181	1538.80	2223.60	125.10	36 / 78

Figure 26: Scenario 3 GAM Results

7.4. Scenario 4

Scenario 4 was similar to scenario 3 except that the direction of movement by the GAF was different. Figure 27 shows the EGAT results for the chosen test positions.

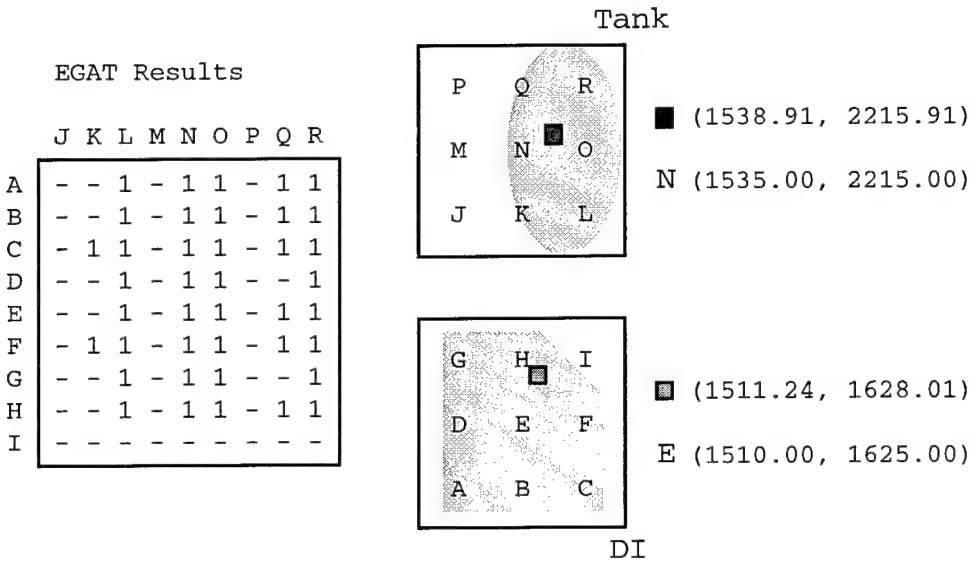


Figure 27: Scenario 4 EGAT Results

Figure 28 shows the entity positions. The DI was near the MOUT area and the tank was 0.6 km away.

Figure 28: Scenario 4 Positions

Figure 29 shows the tank's view of the DI in the tank database. In the DI database, each entity could not see the other. In the tank database, each entity had partial visibility of the other.

Figure 29: Tank's View in Scenario 4

Figure 30 shows that the GAMs were not equal and so the tank host applied the GAF. The resulting delta position was 10 pixels left and 1 pixel up, which had a corresponding (X, Y, Z) of (0.555382, -0.023170, -0.030904). By moving the DI behind the corner of an adjacent building approximately 0.6 meters to the left, the DI was no longer visible. If the building was not nearby, the GAF would have hid the DI by moving it below the terrain, as in scenario 3.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1511.24	1628.01	131.05	0 / 171	1538.91	2215.91	124.28	0 / 85
Tank Host	1511.24	1628.01	131.05	118 / 174	1538.91	2215.91	125.66	49 / 61

Figure 30: Scenario 4 GAM Results

7.5. Scenario 5

Scenario 5 was similar to scenario 3 except that the entities' relative positions had been swapped. The DI was in an area where the terrain differs and the tank was in the MOUT area (where the terrain was identical in the two databases). Figure 31 shows the EGAT results for Scenario 5.

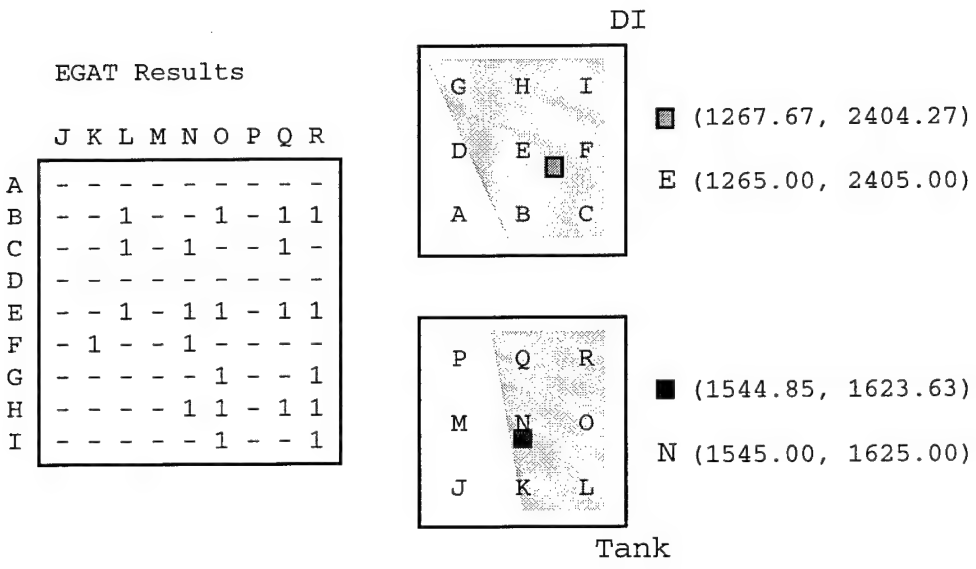


Figure 31: Scenario 5 EGAT Results

Figure 32 shows the entity positions in the databases. The tank was near the MOUT area and the DI was 0.8 km away.

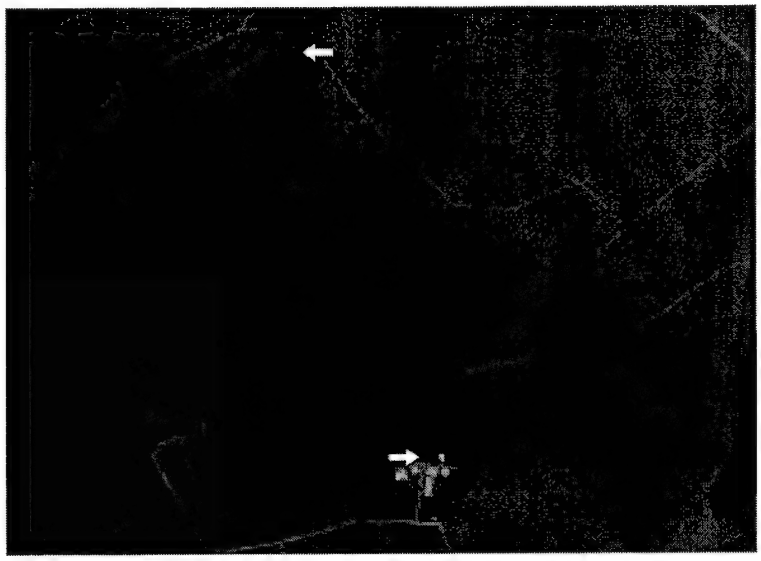


Figure 32: Scenario 5 Positions

In the DI database, each entity could not see the other. In the tank database, the DI was completely visible and the tank was partially visible. Figure 33 shows the tank's view in the tank simulator.

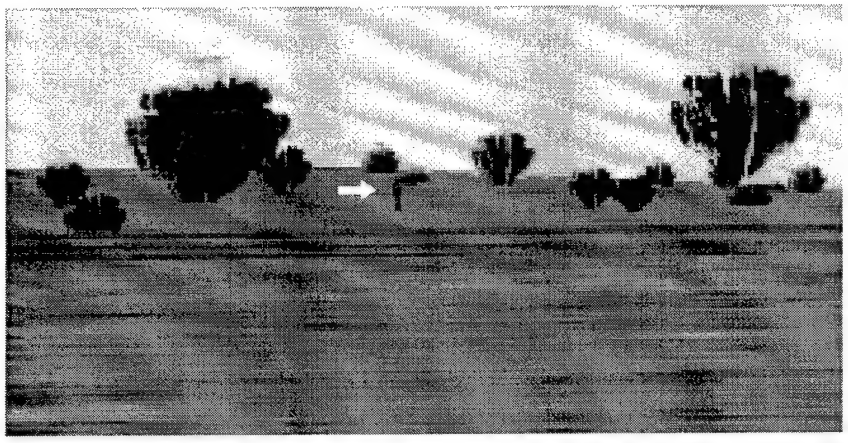


Figure 33: Tank's View in Scenario 5

Figure 34 shows that the GAMs were not equal and so the tank host applied the GAF. The resulting delta position was 4 pixels right and 24 pixels down, which had a corresponding (X, Y, Z) of (0.296706, 0.080683, -1.615814). By moving the DI downward approximately 1.6 meters, the DI was no longer visible.

	DI Position			DI GAM	Tank Position			Tank GAM
	X	Y	Z		X	Y	Z	
DI Host	1267.67	2404.27	120.60	0 / 91	1544.85	1623.63	131.28	0 / 30
Tank Host	1267.67	2404.27	122.00	91 / 91	1544.85	1623.63	131.28	19 / 30

Figure 34: Scenario 5 GAM Results

7.6. Scenario 6

Scenarios 6-8 were related. The DI had the same position near the MOUT area. The tank had positions about 0.5 km away. The tank's positions were along the same line of sight from the DI's perspective, and were successively closer in each scenario. When considered as a sequence, these scenarios will suggest the effect of the algorithms in a continuous, real-time simulation. Figure 35 shows the EGAT results for Scenario 6.

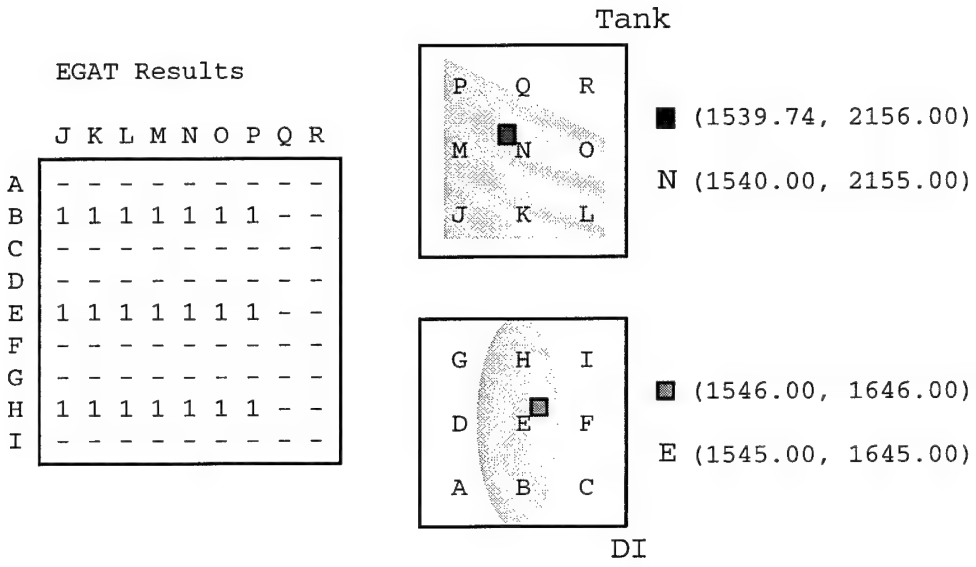


Figure 35: Scenario 6 EGAT Results

Figure 36 shows the entity positions for Scenario 6.

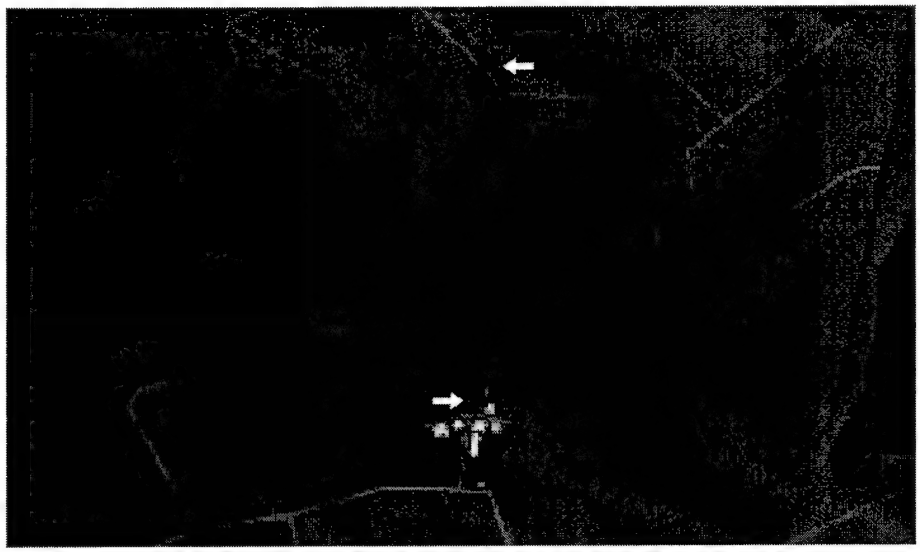


Figure 36: Scenario 6 Positions

The tank had partial visibility of the DI in the DI database and no visibility in the tank database. Figure 37 shows the tank's view in the tank simulator.

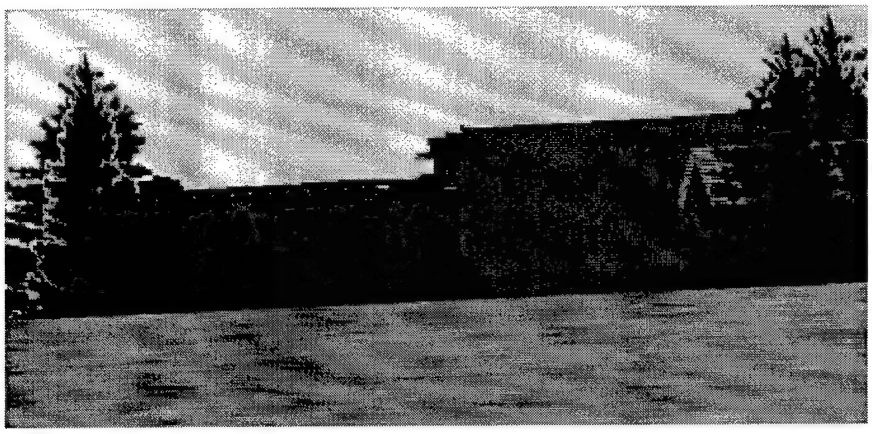


Figure 37: Tank's View in Scenario 6

Figure 38 shows that the GAMs were not equal. The tank host applied the GAF. The resulting delta position was 41 pixels up, which had a corresponding (X, Y, Z) of (-0.091519, 0.018204, 1.687094). After applying the GAF, the DI was moved upwards about 1.6 meters and was almost completely visible.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1546.00	1646.00	131.10	219 / 225	1539.74	2156.00	127.41	57 / 77
Tank Host	1546.00	1646.00	131.10	0 / 237	1539.74	2156.00	125.03	20 / 97

Figure 38: Scenario 6 GAM Results

Figure 37 shows the tank's view of the DI in the tank simulator when the DI's position has been adjusted by the delta position. It appears that the DI is standing on the terrain.

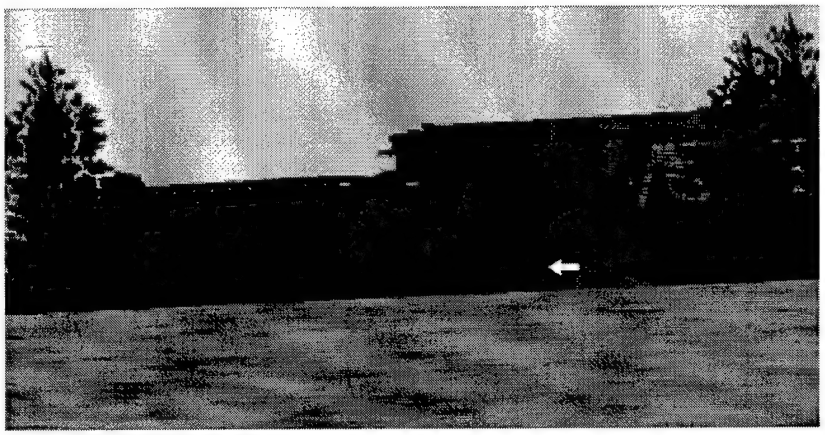


Figure 39: Tank's View of DI Delta Position in Scenario 6

7.7. Scenario 7

In Scenario 7, the tank was moved forward along its heading from the position in Scenario 6 by about 10 meters. The EGAT results were similar to Scenario 6, and are shown in Figure 40.

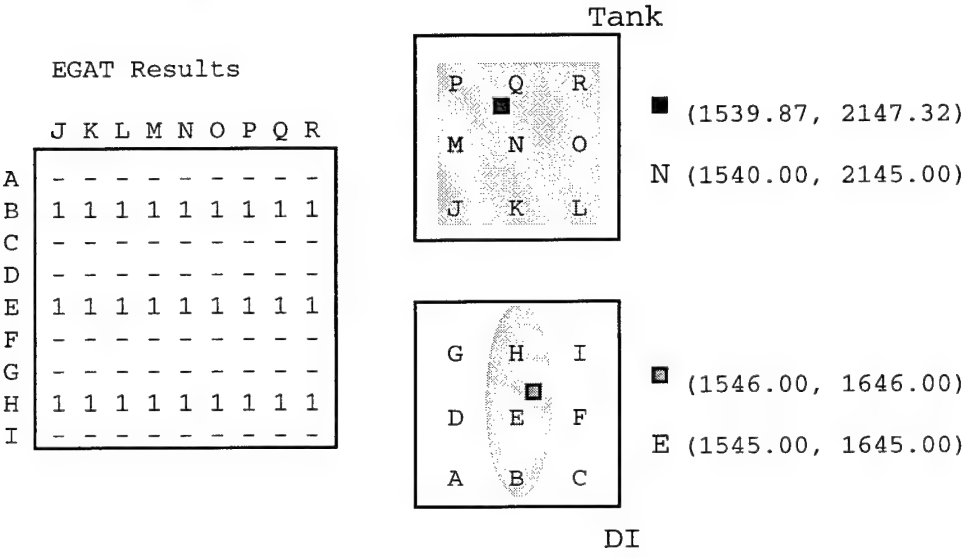


Figure 40: Scenario 7 EGAT Results

Figure 41 shows the entity positions for Scenario 7.

Figure 41: Scenario 7 Positions

The DI was completely visible in the DI database and not visible in the tank database. Figure 42 shows the tank's view in the tank simulator.

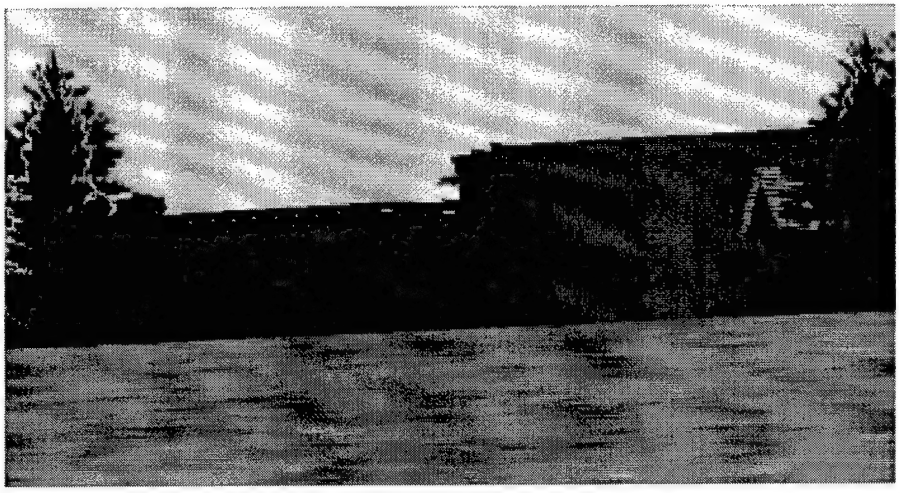


Figure 42: Tank's View in Scenario 7

Figure 43 shows that the GAM results were not equal, and so the tank host applied the GAF. The resulting delta position was 41 pixels up, which had a corresponding (X, Y, Z) of (-0.089799, 0.007689, 1.658102). After applying the GAF, the DI was moved upwards about 1.6 meters. However, the GAM for the delta position differed by more than 33%.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1546.00	1646.00	131.10	245 / 245	1539.87	2147.32	127.77	71 / 98
Tank Host	1546.00	1646.00	131.10	0 / 249	1539.87	2147.32	124.61	6 / 103

Figure 43: Scenario 7 GAM Results

Figure 44 shows the tank's view of the DI in the tank simulator after the GAF delta position had been added to the DI's position. The DI was not completely visible, but the visibility had been increased by the GAF. A larger threshold radius would have allowed the DI to be moved farther. Section 9 will discuss this issue in more detail.

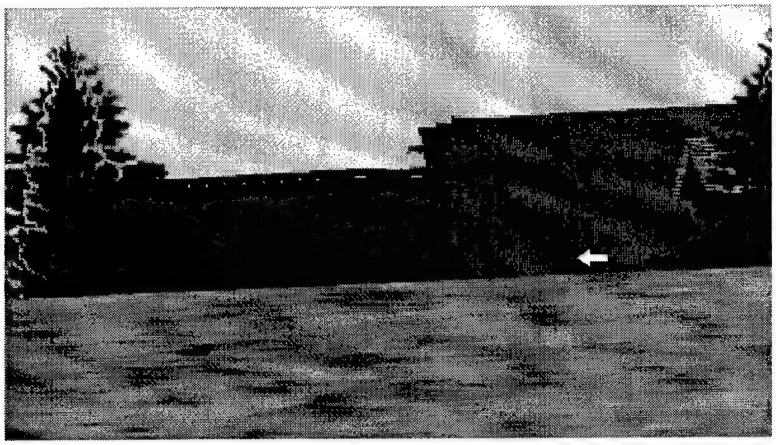


Figure 44: Tank's View in Scenario 7 of DI Delta Position

7.8. Scenario 8

In Scenario 8, the tank was moved forward from the position in Scenario 7 by about 20 meters along its heading. Figure 45 shows the EGAT results for the test positions. As expected, they were similar to the previous two scenarios.

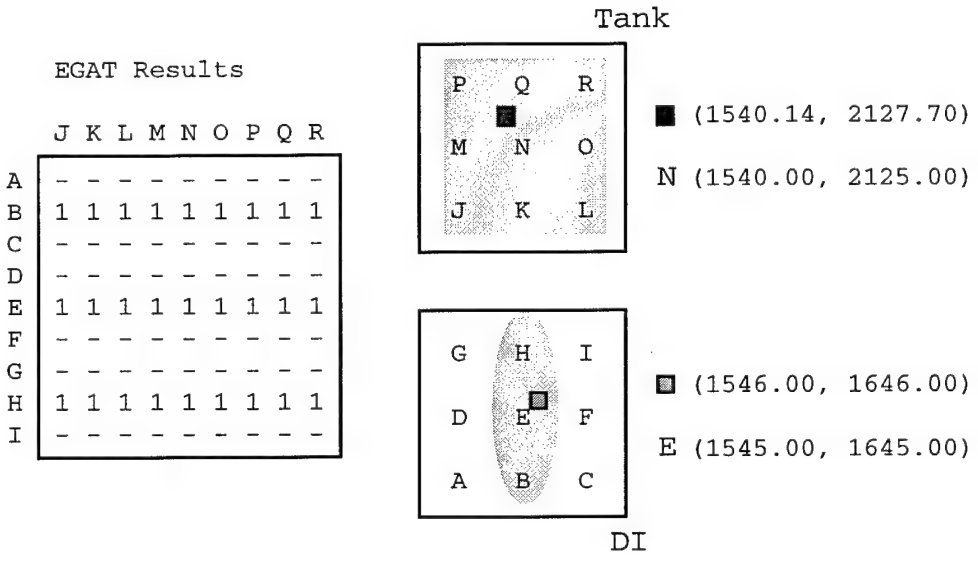


Figure 45: Scenario 8 EGAT Results

Figure 46 shows the entity positions for Scenario 8.

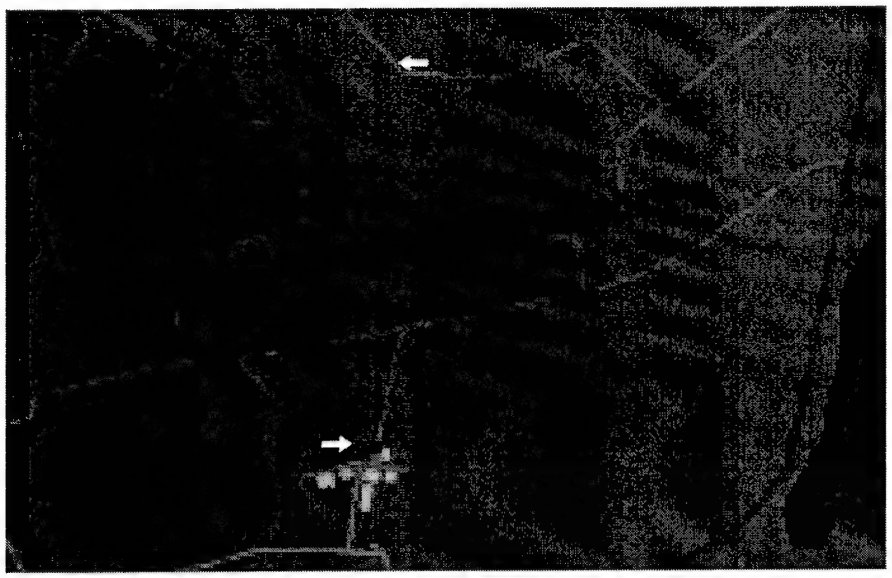


Figure 46: Scenario 8 Positions

The DI was partially visible in the DI database and was not visible in the tank database. Figure 47 shows the tank's view in the tank simulator.

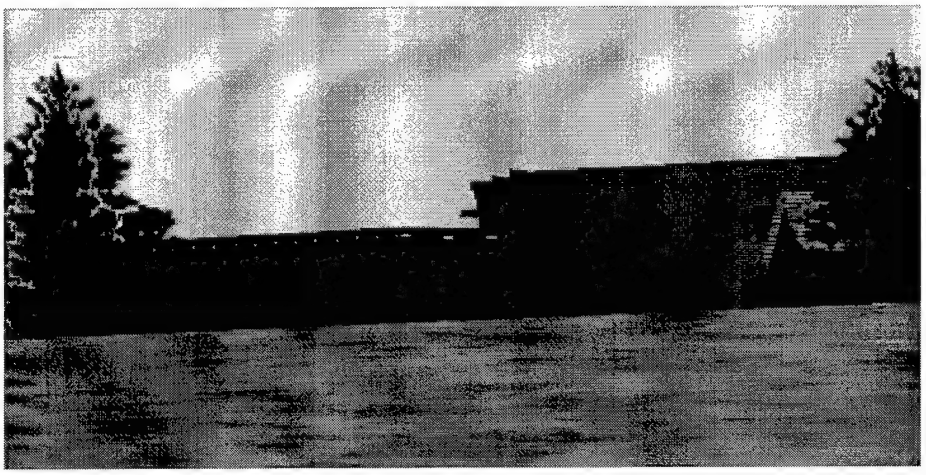


Figure 47: Tank's View in Scenario 8

Figure 48 shows that the GAMs were not equal, and so the tank host applied the GAF. No delta position was possible within the threshold radius. A larger threshold radius was necessary to move the DI beyond the area of occlusion.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1546.00	1646.00	131.10	256 / 263	1540.14	2127.70	127.72	97 / 111
Tank Host	1546.00	1646.00	131.10	0 / 274	1540.14	2127.70	123.65	0 / 118

Figure 48: Scenario 8 GAM Results

7.9. Scenario 9

Figure 49 shows the EGAT results for the test positions. These results were particularly interesting because the EGAT was run after the actual scenario position was found. In this case, the EGAT did not find a VR difference in any areas around the entity positions. The scenario does have an intervisibility problem through. The intervisibility difference is between one viewpoint and parts of the target, which are away from its viewpoint. The description of the EGAT discusses this limitation.

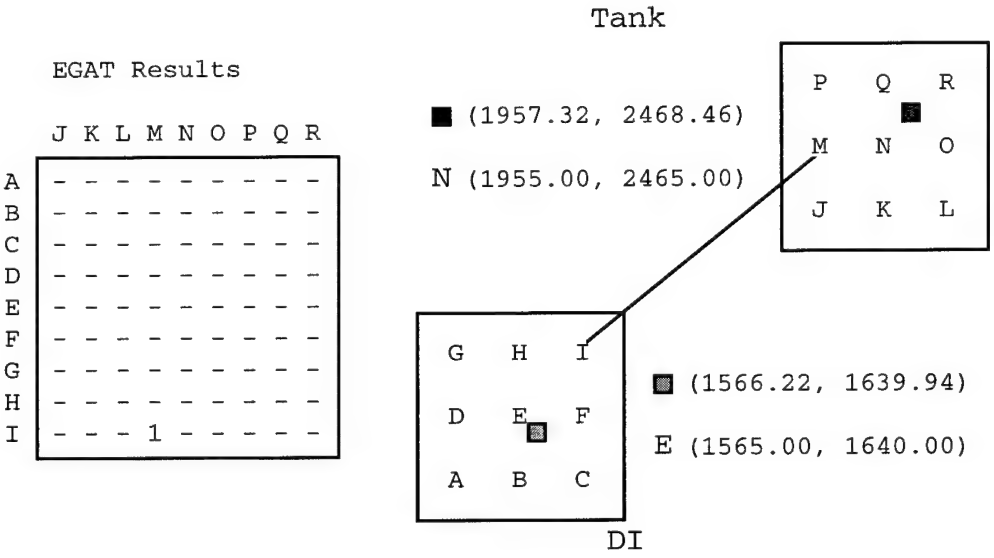


Figure 49: Scenario 9 EGAT Results

Figure 50 shows the entity positions. The DI was near the MOUT area and the tank was 0.9 km away.

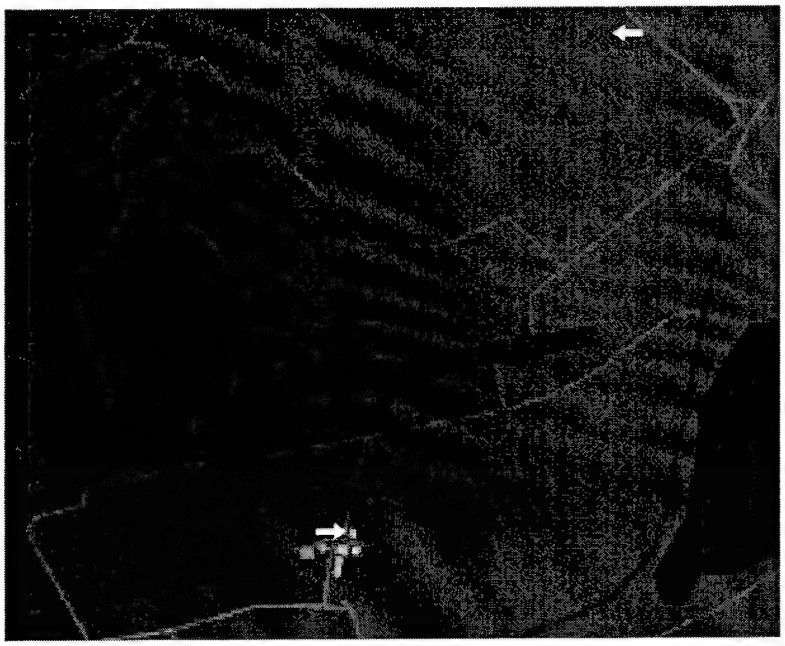


Figure 50: Scenario 9 Positions

In the DI database, the DI was partially visible. In the tank database, the DI was completely visible. Figure 51 shows the tank's view in the tank simulator. In this scenario, the DI was kneeling.

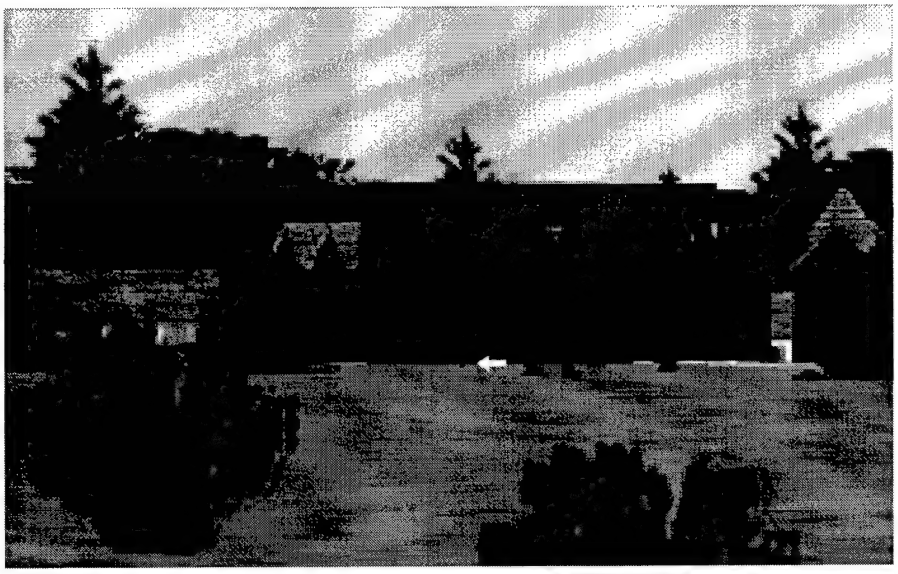


Figure 51: Tank's View in Scenario 9

Figure 52 shows the GAM results. The GAMs were not equal and so the tank host applied the GAF. The resulting delta position was 5 pixels down, which has a corresponding (X, Y, Z) of (-0.002111, 0.003545, -0.372447).

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1566.22	1639.94	130.93	42 / 56	1957.32	2468.46	130.25	21 / 23
Tank Host	1566.22	1639.94	130.93	54 / 56	1957.32	2468.46	130.17	21 / 22

Figure 52: Scenario 9 GAM Results

Offsetting the DI's position by the GAF delta position resulted in the DI being moved downward about 0.4 meters in order to make it less visible. Figure 53 shows an independent view of the DI in the GAF delta position. The penetration of the DI model into the terrain is apparent. This could be unacceptable or could be explained as the result of the lack of detail in the tank database (e.g. the DI is in a trench).

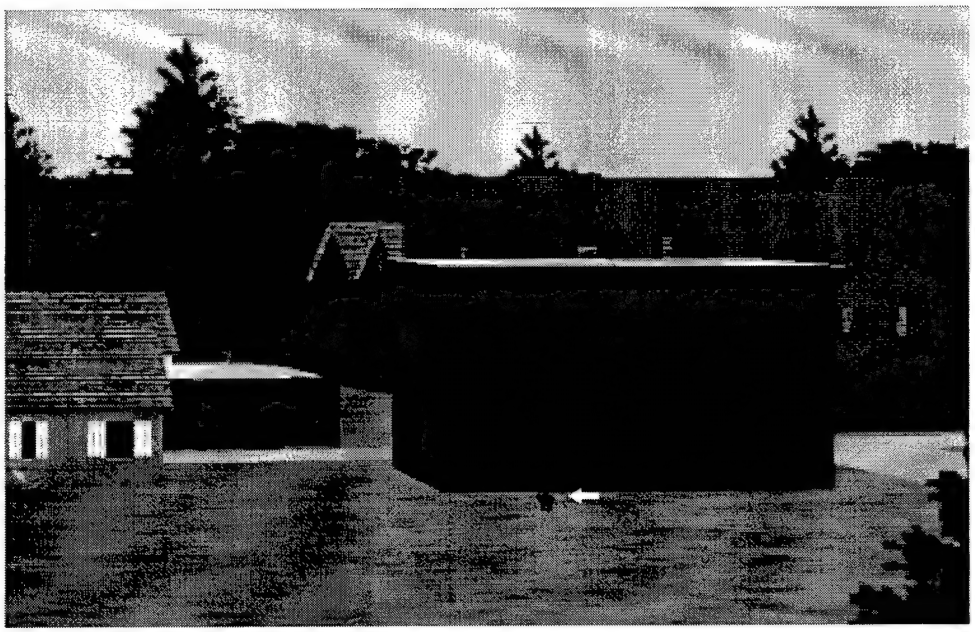


Figure 53: Independent View in Scenario 9 of DI Delta Position

This scenario is an example of a situation in which the GAF may not be able to move the entity to an acceptable position. Such a situation will typically involve an entity being visible on terrain with a face normal pointing toward the viewpoint. If there is no occulting geometry between the two positions and the entity must be made less visible, there will be no alternative besides sinking the entity into the terrain. Extensions to the GAF, such as alternate model representations or occlusion by special effects, may allow more acceptable solutions.

7.10. Scenario 10

Figure 54 shows the EGAT results for the test positions for Scenario 10. This scenario had particular significance because the DI was within one of the buildings. The building's geometry limited to where the DI could be moved to be seen. However, it could help in hiding the DI.

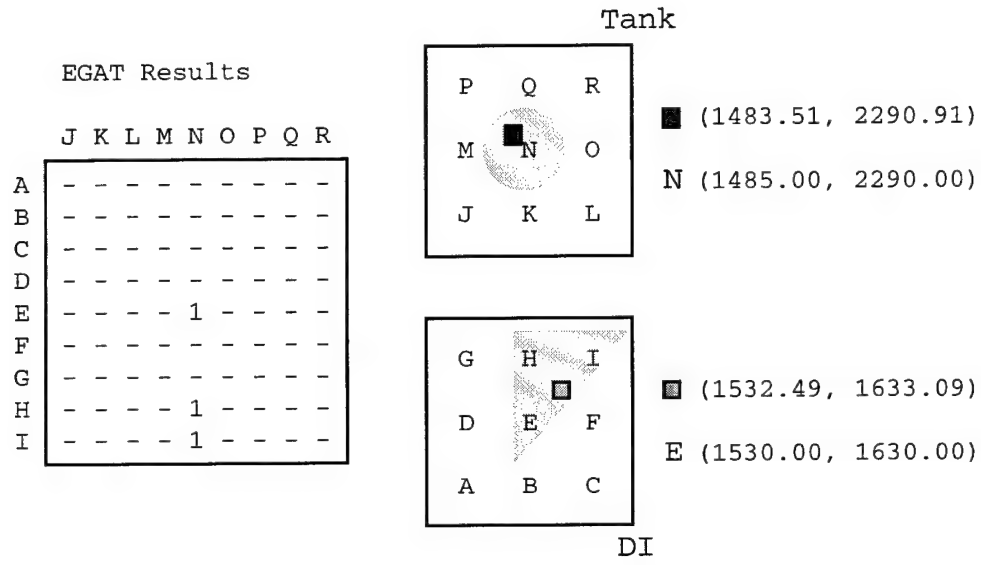


Figure 54: Scenario 10 EGAT Results

Figure 55 shows the positions of the entities in the databases. The DI was in a building looking out of a window and the tank was 0.7 km away.

Figure 55: Scenario 10

In the DI database, the DI was partially visible. In the tank database, the DI was not visible. Figure 56 shows the tank's view in the tank simulator.

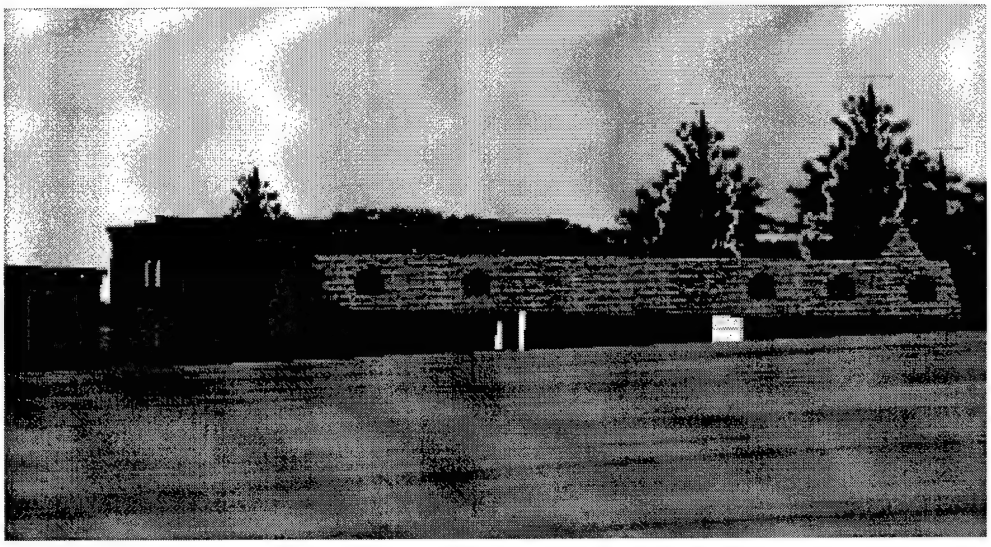


Figure 56: Tank's View in Scenario 10

Figure 57 shows that the GAMs were not equal, and so the tank host executed the GAF. However, no delta position was possible within the threshold radius. Moving the model upward to avoid the occulting of the intervening DBG resulted in the model becoming occulted by the building. Such a situation would be quite common in MOUT area simulations.

	DI Position			DI	Tank Position			Tank
	X	Y	Z	GAM	X	Y	Z	GAM
DI Host	1532.49	1633.09	131.40	104 / 141	1483.51	2290.91	127.52	10 / 59
Tank Host	1532.49	1633.09	131.18	0 / 142	1483.51	2290.91	127.15	4 / 59

Figure 57: Scenario 10 GAM Results

It is worth noting that by increasing the threshold radius, the DI could be made visible by allowing the DI to be moved upward far enough that it is visible through the second story window. While the GAMs might be equal, this would give false information to the tank entity.

8. Other Interoperability Issues

A database for a vehicle, such as a tank, might have details inside of buildings removed for improved simulation performance since the tank cannot enter the buildings. This omission creates problems when opposing entities enter the buildings. For example, if a DI climbs to the second floor of a building, the tank database will not have the floor present. A related concern exists when multiple floors are present since positions inside a building have multiple Z elevations. As an example, the DI in the DI database might be on the second floor of a building. The Z for the ground floor might be 130 meters while the Z for the second floor might be 134 meters. In the tank database, the terrain could be higher and the building floors might have Z's of 133 and 137, respectively. A mechanism to guarantee the proper position may need to be used. Such a mechanism might involve additional information in a PDU.

9. Conclusions

From our experiment, we conclude that altering a remote entity's perceived visibility, by repositioning the remote entity, is effective in many cases. We have shown that when our algorithms succeed, the interaction between the two entities is significantly more fair. However, there are cases when the algorithm's cannot resolve large intervisibilty errors.

Two cases that occurred when the GAF could not resolve the difference in GAMs are (1) the "doorway" scenario, where delta positions are occulted by something other than the terrain geometry, and (2) an insufficient threshold radius scenario, where suitable delta positions are beyond the threshold radius. Case 1 (as shown in scenario 10) may not have any other solutions other than (a) eliminating the terrain geometry difference through database modeling or (b) avoiding the areas which have a high incidence of terrain correlation errors. Case 2 might be alleviated by increasing the threshold radius. However, the computational impact of increasing the sample point quantity may create a processing overload problem on the host simulator.

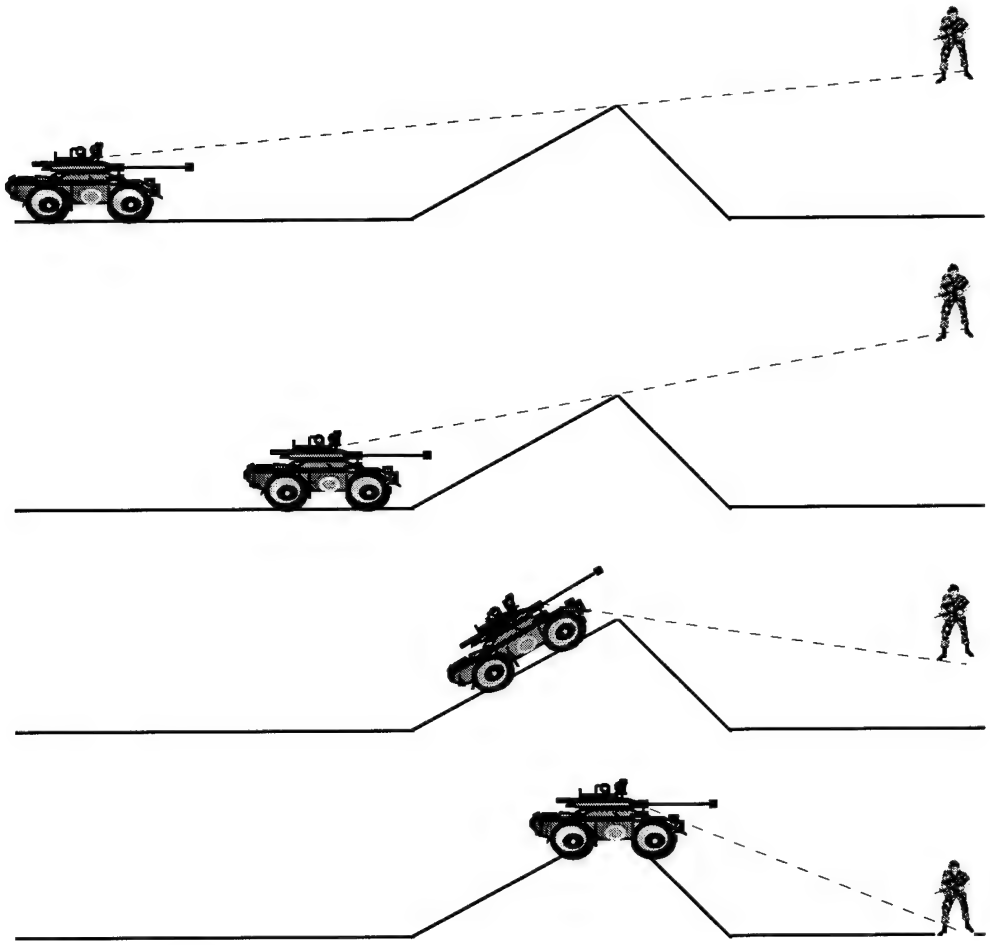


Figure 58: A Sequence of GAF Delta Positions in a Continuous Simulation

An example of the typical result of the GAF algorithm when used in a dynamic environment is illustrated in Figure 58. When the tank entity approaches the remote DI entity, the remote entity is displaced vertically to make it appear to be standing on top of the terrain. As the "hill" is approached, the remote entity is positioned higher until the tank's viewpoint nears the apex. The remote entity would descend to the actual terrain height as the tank comes over the top of the hill. From the tank entity's point of view, the remote DI entity would appear to be standing on the terrain all of the time. In areas with large elevation differences between the two DBG's a better solution may be to eliminate the geometry difference or to avoid such areas.

Our implementation of the algorithms has a processing overhead that is currently too prohibitive to apply in real-time for most simulations. The efficiency of the implementation could be improved, but it is likely that there would still be a significant computational impact to the host computer in real-time. If we factor in

the possibility of executing the algorithms for multiple remote entities, the computational limits are reached even more quickly.

We must also consider the requirement that all simulators interacting in an exercise must have the algorithms implemented. This would have a large spin-up cost, but probably would be reasonable if it were established as a standard.

Despite the limitations, we should still recognize the potential of the approach. The next section will discuss other work which could take advantage of the potential while eliminating most of the drawbacks.

10. Recommendations

A useful extension to the technology developed in this experiment would use the GAM and GAF to analyze databases off-line (similar to the EGAT). Scenario corridors could be examined for VR differences. In many cases, altering an entity's path slightly can reduce or remove the problem.

Since we have concluded that altering an entity's appearance to correct intervisibility interoperability problems is not 100% effective, the alternative stated in Section 3 says that we may need to "modify the DBGs so that the difference does not occur." We have already noted reasons as to why various simulators require different DBGs in order to perform effectively. Also, identical DBG's do not necessarily eliminate intervisibility problems. Therefore, it is not feasible to impose a requirement of identical DBG's as a global solution.

It was observed during the experiment that there are many pairs of positions in the databases which have differences in terrain geometry, yet do not have a difference in intervisibility. For example, although there may be an intervisibility difference due to poor terrain geometry correlation, the introduction of trees or other objects might may allow for a fair fight despite the terrain differences. So, databases do not have to be identical everywhere. This concept allows some freedom to modify a DBG for a particular simulator, yet ensure that it will interoperate with another version of the DBG.

Another off-line tool which could reduce the processing large areas of the database is a process which would be run after the simulation is complete. The entity positions and events of interest could be logged during the simulation exercise. The off-line tool could analyze the interactions between the entities and flag intervisibility errors that may have influenced the outcome of an event. This analysis would be useful as part of an after action review.

When implementing the GAM and GAF algorithms (either as run-time host functions or off-line tools), improvements over our implementations should be considered. Handling translucency and texture, multiple levels-of-detail, and

multiple entities are very desirable extentions. Incorporating any of these extensions should not change the GAM and GAF concepts. Certainly efficiency enhancements to the GAM computations would be beneficial.

Finally, we recommend the development of a test database as aid to future database analysis algorithms. Quite often during the experiment, testing and debugging times were excessive when specific cases had to be found within the McKenna MOUT DB. A generic test database containing specific documented features could be used to attain more predictable results during the initial algorithm development and integration phase. For the final integration and test the McKenna MOUT DB was a useful environment.

11. Appendix

The figures in the appendix were generated from output during the execution of scenario 4. Other scenarios generated similar data and they have been omitted. Masks have the rows numbered bottom to top with 0 at the model's center. The columns are numbered left to right with only the least significant digit printed above each column and with 0 at the model's center.

Model mask entries have two possible values. An asterisk '*' denotes that the model was hit by the sample. A space ' ' denotes that it was not. Occult mask entries have four possible values. A space ' ' denotes that nothing was hit by the sample. A period '.' denotes that a sample was not taken. An asterisk '*' and a tilde '~' both denote that a database face was hit by the sample. Entries marked with a tilde are supposed to denote a terrain intersection. Since there was no way to identify this in the DBI, an intersection was assumed to be terrain if the face normal pointed roughly upward (i.e., had a Z component > 0.7). This was not necessary for the GAM computations, but aided in the analysis of the output.

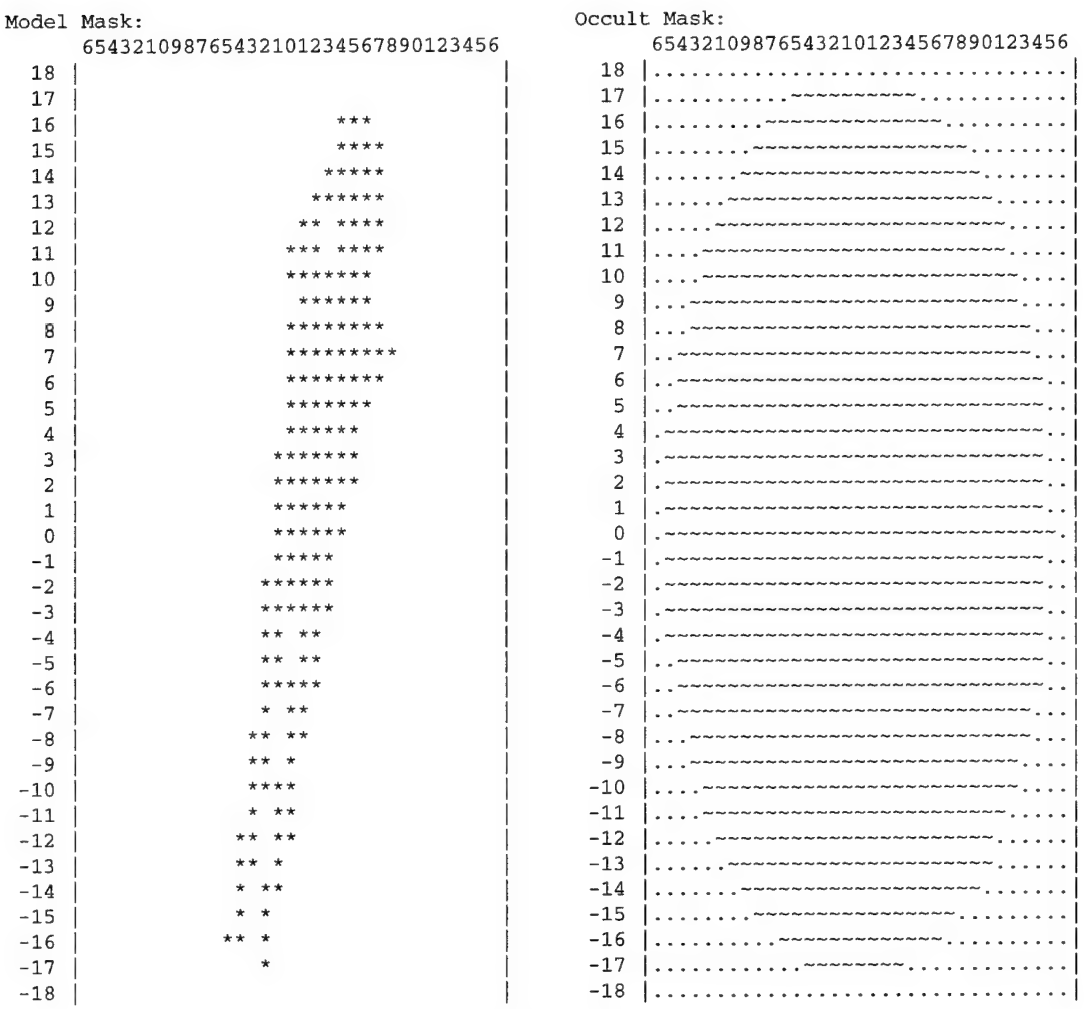


Figure 59: Scenario 4 - DI Host - DI Masks

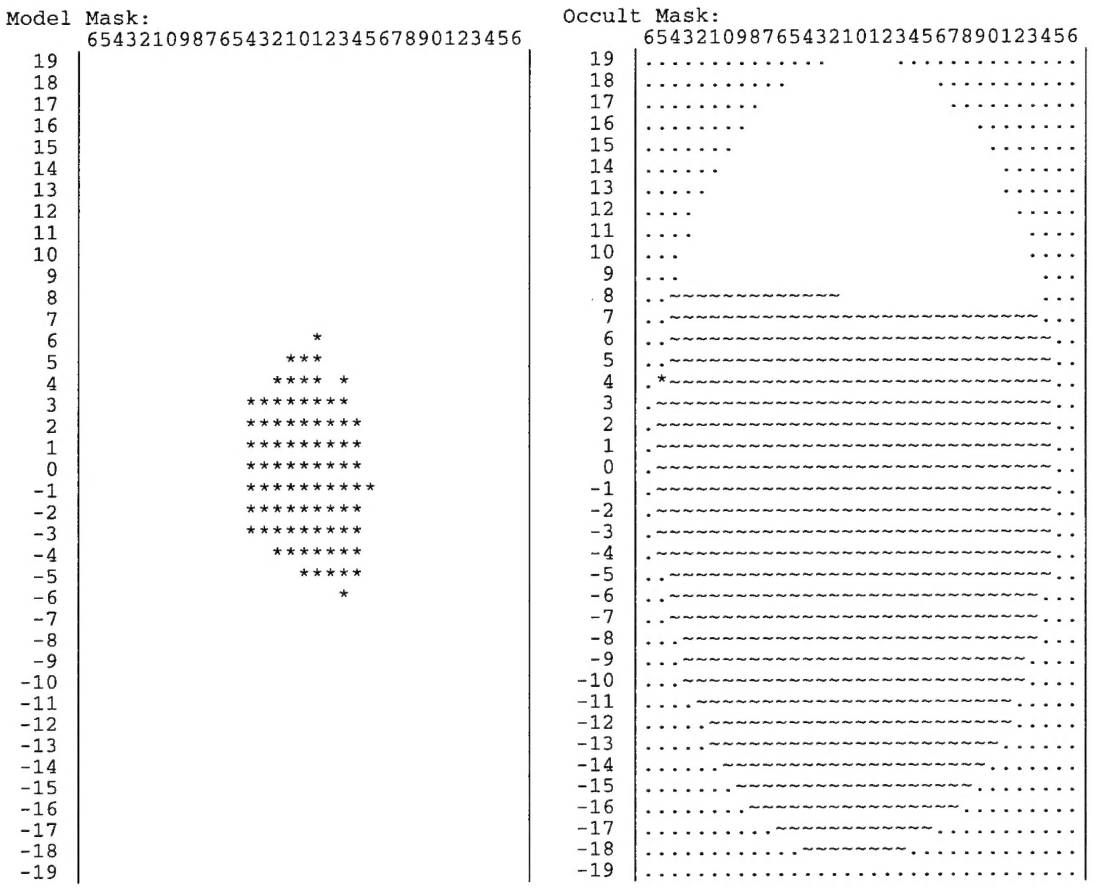


Figure 60: Scenario 4 - DI Host - Tank Masks

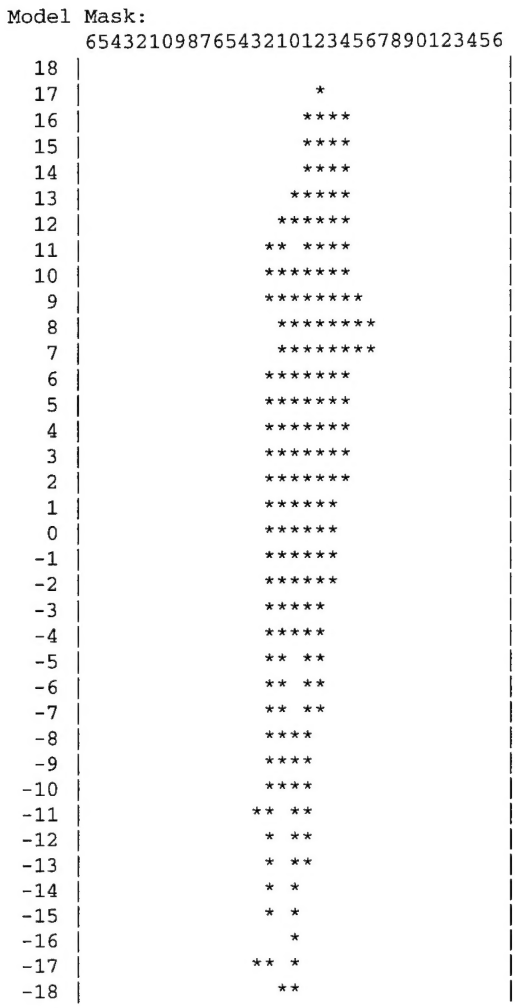


Figure 61: Scenario 4 - Tank Host - DI Model Mask

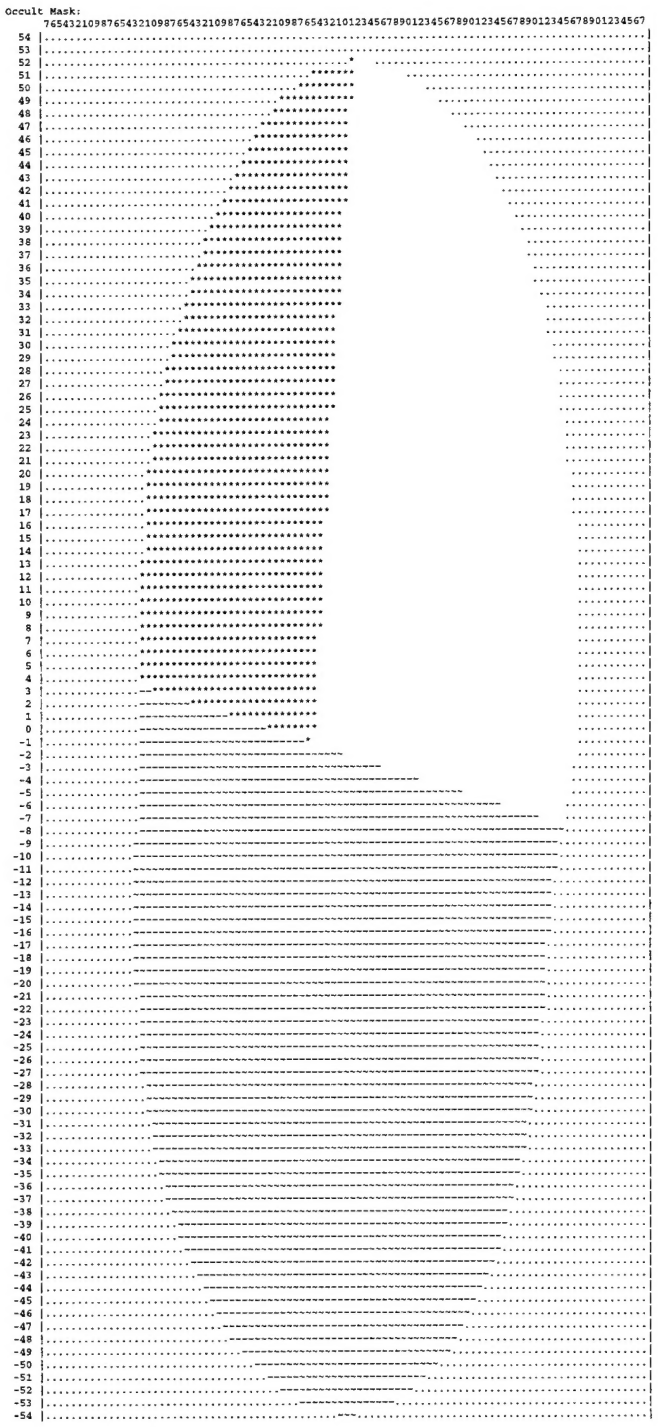


Figure 62: Scenario 4 - Tank Host - DI Occult Mask

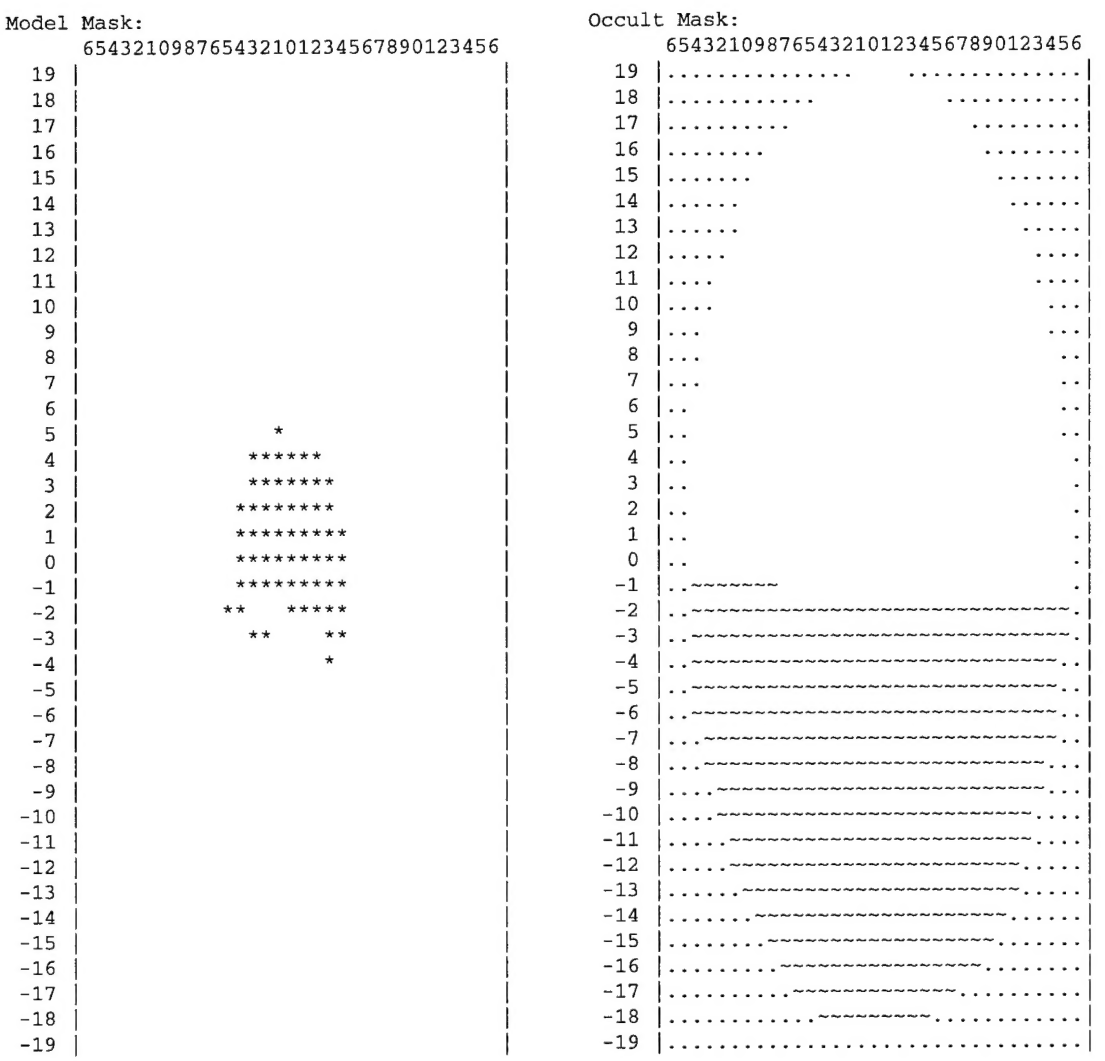


Figure 63: Scenario 4 - Tank Host - Tank Masks